dreamtech PRESS

- An all-inclusive book to teach you everything about Developing BizTalk 2006 Applications
- Easy, Effective, and Reliable
- Quick and Easy learning in Simple Steps
- Most preferred choice worldwide to learn how to develop BizTalk 2006 Applications

Developing BizTalk 2006 Applications

IN SIMPLE STEPS

Easy to learn with hundreds of illustrations.

Do it right, Do it fast!

Developing BizTalk 2006 Applications

IN SIMPLE STEPS

Developing BizTalk 2006 Applications

IN SIMPLE STEPS

Authored by:

Kogent Solutions Inc.

Published by:

dreamtech PRESS

ISBN: 10-81-7722-857-9

13-978-81-7722-857-1

Edition: 2008

Printed at: Printman India, Patparganj, Delhi.

CONTENTS

Chapter 1

Getting Started with BizTalk Server 2006

In this Chapter

- An Introduction to BizTalk Server
- What is BizTalk Server 2006?
- BizTalk Server Architecture
- Deploying BizTalk Server 2006
- Configuring BizTalk Server
- Installing Enterprise Single Sign-on
- Installation of SSO Client Utility
- Tool in BizTalk Server 2006

BizTalk Server 2006 (BZ) is an integration server, based on .Net technologies. BizTalk Server helps you to integrate different applications across multiple organizations and perform B2B (business-to-business) transactions, using the Internet. It uses third party adapters to perform integrations with third party B2B systems. It enables companies to integrate and manage business processes by exchanging business documents, such as purchase orders and invoices, among distinct applications, within or across organizational boundaries.

BZ 2006 exchanges information by sending and receiving messages in XML format among different organizations. This helps in centralizing the exchange of information from source to destination.

This chapter introduces the latest version of BizTalk Server, known as Microsoft BizTalk Server 2006 or simply BZ 2006, and describes its features in detail. It also explains the basic concepts related to BizTalk Server 2006, the various types of architecture of the server; and finally, how to install and configure BizTalk Server 2006.

An Introduction to BizTalk Server

Microsoft BizTalk Server, also known simply as "BizTalk", is a business process solution server. It provides the infrastructure and tools for building successful business-to-business solutions, business process automation, business process modeling, business-to-business communication and enterprise application. Although its targets mostly medium and large businesses, BizTalk has also become popular with small companies.

A Business Process Management (BPM) server, BizTalk helps companies to automate and optimize business processes and provide powerful tools to design, develop, deploy and manage the business process. Using BizTalk, developers, IT professionals, and business analysts can easily build dynamic business processes spanning applications over the Internet. With BizTalk, companies need to use fewer resources to get their solutions to the market quickly in response to customer needs and competitive pressures.

Microsoft BizTalk Server provides a powerful Web-based development and execution environment that integrates long-running business processes between businesses activities and transactions that used to take months to complete but now take just a few minutes. The BizTalk framework provides a standard gateway for sending and receiving documents across the Internet, as well as providing a range of services that ensure data integrity, delivery, and security.

What is BizTalk Server 2006?

BizTalk Server 2006 is the latest release of BizTalk Server developed by Microsoft. BZ 2006 provides support for building solutions for business process and integration. BZ 2006 is the fourth version of BizTalk Server from Microsoft technology. Earlier versions of BZ are 2000, 2002, and 2004.

BZ 2006 is built on .NET framework 2.0 and its developer tools are hosted in Visual Studio 2005. It uses SQL Server 2005 for storing database information. It is also supported on 64-bit Windows operating system, thereby taking the advantage of a larger memory and other benefits, such as security and compatibility. Some of the newly introduced features of BZ 2006 are as follows:

- BZ 2006 provides better and easy support for deploying, monitoring, and managing applications.
- BZ 2006 applications are considerably easier to install as compared to previous versions.
- BZ 2006 interface is simpler than that of BizTalk Server 2004.
- BZ 2006 provides improved capabilities and enhanced functions for Business Activity Monitoring (BAM). BAM is a collection of tools that allows you to create, deploy, and view information on running business processes. It gives you the status and results of the various operations, processes and transactions of your business, along well as the problem areas. This gives you the opportunity to take timely remedial action wherever needed, as soon as possible.

Combining different systems into effective business processes is a very challenging task; however, this can be achieved very easily by using BZ 2006. Fig.Biz-1.1 illustrates the major components of BZ 2006.

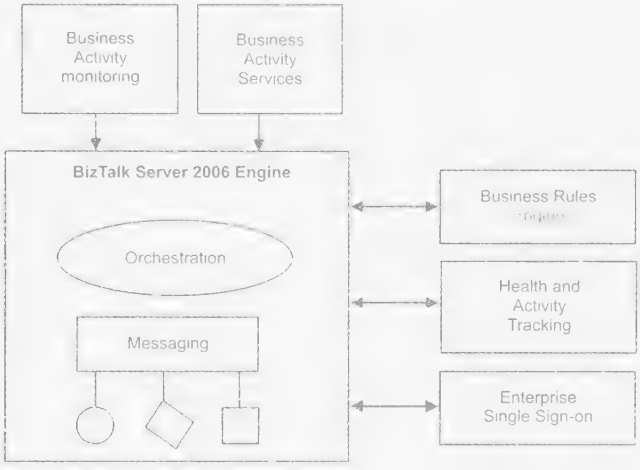


Fig.Biz-1.1

The figure shows how the BZ 2006 engine performs various business processes by sending and receiving messages among different components of BZ 2006. The BZ 2006 engine contains two major parts:

- A messaging part that is used for communicating with other applications. The messaging is done for different kinds of communications through adapters such as FILE, FTP, HTTP, POP3, and SMTP. The BZ 2006 engine supports various protocols and data formats, including Web services.
- The orchestration part, which is used to build graphically defined business processes through a sequence of diagrams. Using such diagrams makes a business process easier to understand than if it were explained in code. You can orchestrate customized business processes within organizations with the help of developers, information workers, and IT professionals who can work together to define, design, and deploy integrated solutions that work across applications, platforms, and organizations.

Several other components can also be used with the BZ 2006 engine, such as:

- A Business Rules Engine is required to create rules that are used to determine an appropriate business action in .NET objects and XML documents.
- A Health and Activity Tracking (HAT) tool that enables developers and administrators to monitor and manage the server engine and orchestrations so that they run properly. It can also be used to solve business activity errors by those users who have no deep technical knowledge of BZ 2006.
- An Enterprise Single Sign-on (SSO) utility that provides the ability to map authentication information between Windows and non-Windows systems. With this component, the user can easily log on to multiple services with a single login.

So now you are aware of the BZ 2006 and its components. Let's move further to have a detailed overview of the features of BZ 2006. To know more about the HAT tool, please refer to Chapter 7.

Features of BizTalk Server 2006

BZ 2006 has several key enhancement features that build on the core architecture of BizTalk Server 2004. These feature deal with application-to-application, business-to-business, and business process automation.

BZ 2006 comes with improvements in developer tools to simplify the development of complex applications. Some of the new and enhanced features of BZ 2006 are as follows:

- **Simplified Setup Configuration:** BZ 2006 provides a faster, easier and simplified installation wizard. The user can simply *click* the configuration wizard to configure BZ 2006. Some new and enhanced features available when setting up and configuring BZ 2006 are as follows:
  - Simplified one-computer installation.
  - BZ 2006 comes with component updater technology that checks for missing components during installation and installs them automatically from the Internet.
  - Supports flexible installation on multi-computers.
  - Simple upgradation from BZ 2004.
  - Can be installed on 32 and 64-bit computers, and automatic installation for 64-bit software on 64-bit operating systems.
  - Default configuration setup for new developers.
- **Better Management and Operations Supports:** The various enhancements to improve management and operations related processes help the IT administrator to be more productive in enterprise deployment. The new and enhanced features available in this area are:
  - BZ 2006 provides BizTalk Server 2006 Administration Console through which you can configure, deploy, manage and monitor your BZ 2006 applications easily.
  - BZ 2006 provides a task-based design for easy development and access to frequent and common tasks.
  - BZ 2006 provides the functionality of performing root-cause analysis by powerful and simple querying.

- BZ 2006 provides better and easier support for remote configuration and administration of multiple BZ 2006 groups.
- BZ 2006 provides an enhanced error and event reporting tool (Health and Activity Tracking), which includes an exception message box.
- BZ 2006 provides simplified deployment, better management, and easy troubleshooting for BizTalk Applications.

- **Business Activity Monitoring (BAM) and Business Activity Services (BAS) Facilities:** BAM allows you to create, deploy, and view information about running or previously run processes. BAM provides real time information about the status and results of various operations, processes, and transactions in the organization. BAS is basically a series of services that are hosted on Windows SharePoint Services to help the business user to work with a trading partner. Each trading partner that you want to set up will need an agreement in BAS. Once the agreements are created and deployed in BAS, you can handle business activities very easily with the trading partners. Windows SharePoint Services is used to create Web sites for information sharing and document collaboration benefits to increase team productivity.
- **New and Enhanced Adapter:** BZ 2006 comes with new tools and adapters such as the Flat File Schema Creation Wizard and the Windows SharePoint Services adapter respectively. The Flat File Schema Creation Wizard is discussed in Chapter 4. BZ 2006 provides many new and enhanced adapter features, such as:
  - New adapters such as the POP3 receive adapter, and Windows SharePoint Services adapters have been added in BZ 2006.
  - Enhanced adapters such as the Message Queuing adapter, the IBM MQSeries adapter, the File adapter, and the HTTP adapter have been updated in BZ 2006.

After going through the features of the BZ 2006 let's move further to have a clear understanding of the BizTalk Server Architecture.

BizTalk Server Architecture

The architecture of BZ 2006 is shown in Fig.Biz-1.2. As you can see from the figure, the Receiving Port on the left consisting of adapters and the End Point Manager (EPM). EPM is responsible for the communication using the messaging service in BZ 2006, whereas the Receive Pipeline in EPM receives XML documents using the BizTalk Adapters. The Receiving Port is responsible for bringing messages into BizTalk from a source system and storing them in the BizTalk **MessageBox**. BizTalk **MessageBox** also uses the Orchestration, which is the heart of BZ 2006. It is used to process and retrieve messages, required for performing business transactions Any messages that are used by orchestrations are stored in the BizTalk **MessageBox**, from where they are retrieved and sent as and when required.

The Send Port on the right side consists of adapters, the End Point Manager (EPM), and the Send Pipeline. The Send Pipeline is responsible for collecting outbound messages from the BizTalk **MessageBox** and transmitting them to the destination system. The Send Pipeline assembles, encodes, and encrypts messages into an XML document, and then sends them through the BizTalk adapter.

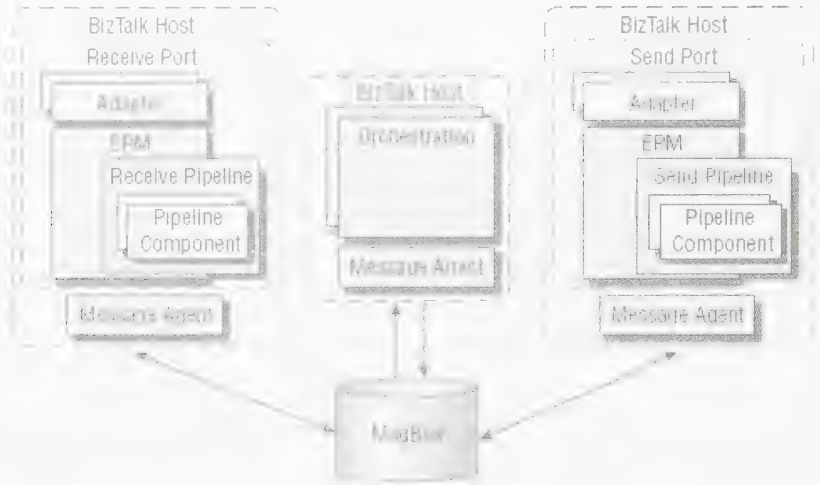


Fig.Biz-1.2

This is the common architecture of BZ 2006. Now that we have discussed the various components of BZ 2006, it is time to learn something about the various types of architecture in BZ 2006. BZ 2006 has four types of architectures based on different components of BZ.

- Runtime Architecture
- Management and Tracking Architecture
- Business Activity Service Architecture
- EDI and AS2 Architecture

Let's begin our discussion with BizTalk runtime architecture.

Runtime Architecture

Runtime architecture of BZ 2006 is built on a publish subscribe architecture in which a message is published into the system, and then received by one or more active subscribers. This architecture is also known as content-based publish/subscribe architecture. In a content-based publish/subscribe model, subscribers specify the messages that they want to receive using a set of criteria about the message. This message is evaluated at the runtime and all of the active subscribers with matching subscriptions receive the message.

Management and Tracking Architecture

The Management and Tracking architecture of BZ 2006 includes components such as Business Activity Monitoring (BAM), planning for Health and Activity Tracking (HAT) and Record size in tracking database. Fig.Biz-1.3 depicts the BAM tool of Management and Tracking architecture of BZ 2006. Let's look at the various components of Management and Tracking architecture in detail.

1. **Business Activity Monitoring (BAM):** The BAM architecture is divided into four broad sections. These sections are Tools, Presentation, Processing and Database. Let's briefly look at these sections.

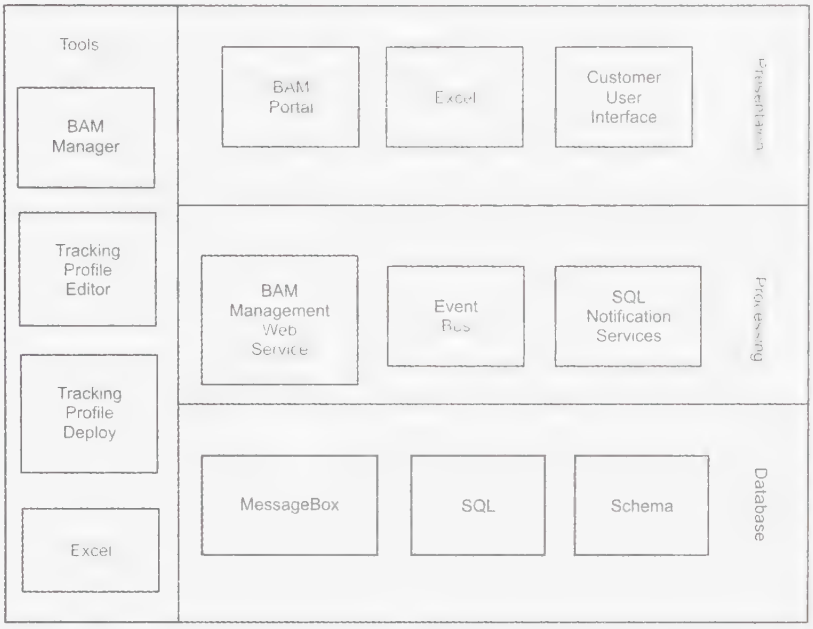


Fig.Biz-1.3

- **Tools:** The following tools are integrated with BAM and are used to design, develop, and deploy BizTalk solutions.
  - **BAM Manager:** This is a deployment tool that allows the Excel tool of BAM to be deployed into an enterprise.
  - **Tracking Profile Editor:** This tool enables BizTalk developers to map the data elements into BZ 2006 and implement these elements in orchestrations and messaging.
  - **Tracking Profile Deploy:** This tool allows users to deploy new and updated schema based messages into the BAM infrastructure.
  - **Excel**: This Excel tool provides a simple user interface that guides business analysts during the creation of business activities.
- **Presentation:** The presentation layer of BAM architecture consists of the following parts:
  - **BAM Portal:** The BAM portal tool provides real-time, end-to-end visibility into a business process. It allows you to perform data searches and aggregate data on your business process. It is a Web-based tool that consists of a collection of ASP.NET 2.0 pages.
  - **Excel:** It provides a user interface that helps and guides business analysts during the creation of business activities in the organization. Excel serves as both a design tool for business analysts and a data consumption tool for business users.
  - **Custom User Interface**: Developers can use this interface to create custom applications that display BAM data.
- **Processing:** The processing layer of BAM architecture consists of the following tools:

- **BAM Management Web Service**: This web service is used with BAM Portal to help clients get related business activities in the organization.
- **Event Bus:** The BAM Event Bus tool processes the tracking data stored in a source database and persists that data in a query table format in the destination database.
- **SQL Notification Services:** This tool generates and sends notification messages during the development and deployment of applications. For example, subscribers might express the following preferences: "Notify me when the stock price reaches Rs.150," or "Notify me when the price of stock in the database is updated."

- **Database:** The database tool is used to store the data and is connected with the processing layer.

Now that you are familiar with BAM components of Management & Tracking architecture, let's move on and learn something about the next component of Management & Tracking architecture: planning for Health and Activity Tracking.

2. **Planning for Health and Activity Tracking (HAT)**: HAT is a monitoring tool used to monitor the BZ 2006 application to solve the errors of your business activities. Here, you can decide and set the selected messages that you need to track during the planning stage because when you track all messages through HAT, it minimize the performance of your system. To know more about HAT, please refer to Chapter 7.

The next component of Management and Tracking architecture is record size in tracking database. Let's look at what this component does in detail.

3. **Record Size in Tracking Databases:** The record size for various event records in the tracking databases is needed when you plan your hardware requirements for BizTalk. For example, to deploy a service, you need 1864+ symbolic information size (1 symbolic information is equal to an orchestration needing approximately 4000 bytes). A service to be successfully completed needs 252 bytes. Start and Call shapes (e these shapes will be discussed in Chapter 2) in an orchestration requires 120 bytes, and a minimum of 162 bytes is needed for incoming and outgoing messages.

You are now familiar with the architecture of Management & Tracking, which is completely based on BAM. Next, we take up another architecture related to BAM, known as Business Activity architecture or BAS architecture.

Business Activity Services Architecture

In business processes, a business analyst may need to create a relationship with a new trading partner. This relationship may be based on, for example, the partner's role, the business agreement between the two firms, and other aspects of this new association. A purchasing manager needs services that he can use to distribute the business activities required in a business process. These services are provided by Business Activity Services (BAS) in BZ 2006.

The BAS architecture is shown in Fig.Biz-1.4. BAS provides a common user interface that enables the business user to use services such as Microsoft Windows SharePoint, Internet Explorer, Microsoft Excel, and Microsoft InfoPath. Behind the common user interface services are two software components. These are Trading Partner Management and Business Process Configuration.

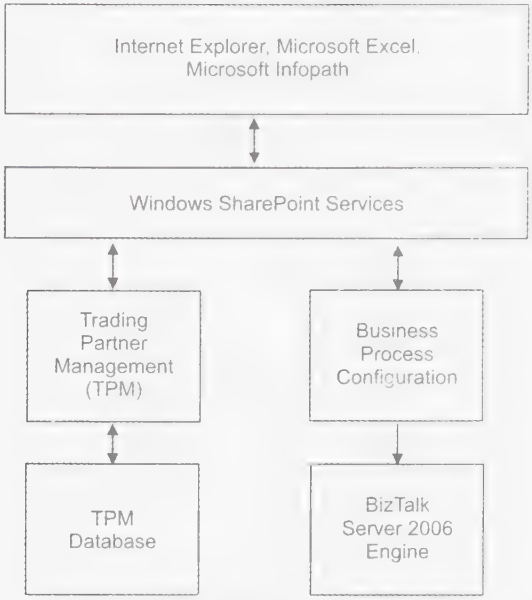


Fig.Biz-1.4

- **Trading Partner Management (TPM):** The TPM component enables you to define user and organization profiles such as contacts, addresses and preferences. It is also used to define business partner relationships, such as buyer/seller, through business agreements. It connects you to your business partners to improve the movement of goods through your facilities and throughout the entire supply chain.

The TPM partner stores information about trading relationships in TPM database, as show in Fig.Biz-1.4. It is also used to store agreements and the business profile of the trading partner.

- **Business Process Configuration:** The Business Process Configuration services allow business users to configure and manage orchestrations that implement the business process. Once the configurable business processes are available, business users can deploy them by using services such as Microsoft Windows SharePoint, Internet Explorer, Microsoft Excel, and Microsoft InfoPath to meet the needs of the organization. Microsoft InfoPath is a part of Microsoft Office 2007 used to develop XML documents

With this, we come to the last two architectures of BZ 2006, called Electronic Data Interchange (EDI) and Applicability Statement 2 or AS2 solution architectures, these are explained in the next section.

EDI and AS2 Solution Architectures

In BZ 2006, Electronic Data Interchange (EDI) is a process of exchanging business documents between companies, whereas Applicability Statement 2 (AS2) is an EDI specification used for data transport onto the Internet by using HTTP protocol. BZ 2006 uses EDI and AS2 solution architectures to provide supply-chain business processes to business partners. Let's now look at these architectures in detail.

EDI Solution Architecture

EDI or Electronic Data Interchange is a process by which business entities exchange data electronically. EDI messaging protocols ensure that data always arrives in a specified format. However, if the data is not in a specified format or is incorrect, it is automatically detected and reported to the business user. EDI uses the message syntax format and the standard format, including ANSI X12 format. The ANSI X12 format is used to develop uniform standards for inter-industry electronic interchange of business transactions. BZ 2006 processes EDI messages by using the Receive and Send pipelines that parse and serialize these messages. Fig.Biz-1.5 shows the exchange of EDI messages from Organization 1 to Organization 2.

Fig.Biz-1.5

When an EDI message is sent by BZ 2006, the EDI looks for the XML schema, validates the message, sends an acknowledgment (if needed), and serializes the EDI batch. The EDI assembler is used to perform the EDI message processing with the help of EDI Send Pipeline.

AS2 Solution Architecture

AS2 or Applicability Statement 2 is a standard by which users transfer EDI data or other data, such as Extensible Marking Language (XML) documents over the Internet by using the HTTP or HTTPs protocol. When a user receives an AS2 message, he or she sends an acknowledgment message to the sender. AS2 can be used to transport EDI or non-EDI messages. For this reason, AS2 processing is designed and configured apart from EDI processing. Fig.Biz-1.6. shows the exchanging of AS2 messages through HTTP protocol from retailer to supplier.

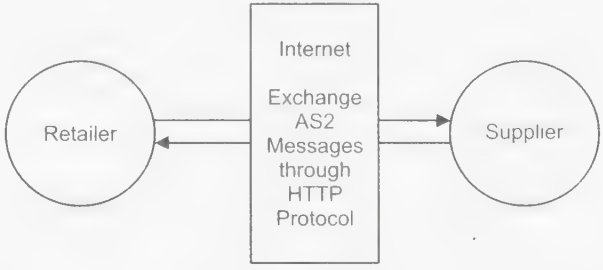


Fig.Biz-1.6

In BZ 2006, AS2 messaging is performed in MIME format of HTTP with AS2-specific headers. The nature of the message depends on whether the message is encrypted, signed, or compressed. If the message is signed, a signature wrapper message is added to the document payload; if the message is compressed, a compression wrapper message is added to the document payload; and if the message is encrypted, an encryption wrapper message is added to the document. BZ 2006 uses the following AS2 pipelines to send and receive AS2 messages:

- **AS2Receive Pipeline**: This pipeline is used to processes messages received over AS2 when the messages are not encoded in EDI format.

- **AS2Send Pipeline**: This pipeline is used to send messages over AS2 when the messages are not encoded in EDI format.

Now that we have familiar with various architecture available in BZ 2006, it is time to move to the next stage and learn how to deploy BZ 2006.

Deploying BizTalk Server 2006

Microsoft BZ 2006 can be installed through simple, easy-to-follow steps. However, before installing the server, you need to first check whether you have the requisite hardware and software for beginning the installation process. Let's look at these requirements.

Hardware Requirements

The minimum hardware needed to install BizTalk in 32-bit systems is as follows:

- **Processor:** Intel Pentium processor with 1 GHz or higher for a single processor, 900 MHz or higher for a double processor and 700 MHz or higher for quad-processors.
- **Memory:** 1 GB of RAM.
- **Hard disk:** Minimum 6 GB of free hard disk space.
- **Drive:** CD-ROM or DVD-ROM.
- **Monitor:** Super VGA with a 1024 x 768 or higher resolution monitor supporting 256 colors or more.

The minimum hardware needed for 64-bit systems is as follows:

- **Processor**: AMD Opteron, AMD64, Intel XEON EM64T, or compatible 1.7 GHz or higher processor recommended.
- **Memory**: 1 GB of RAM.
- **Hard disk**: Minimum 6 GB of free hard disk space.
- **Drive**: CD-ROM or DVD-ROM recommended.
- **Monitor**: Super VGA with a 1024 x 768 or higher resolution monitor supporting 256 colors or more.

Let's now look at the software requirements.

Software Requirements

The minimum software needed to install BZ 2006 on a single computer is as follows:

- Microsoft IIS 6.0 or later versions.
- Microsoft Windows Server 2003 with Service Pack 1.
- Microsoft Office Excel 2003 and InfoPath 2003 with Service Pack 2.
- Microsoft Visual Studio 2005 with Microsoft Visual C# .NET.
- Microsoft SQL Server 2005 or Microsoft SQL Server 2000 with Service Pack 4.
- Microsoft SQL Server 2005 Analysis Services or Microsoft SQL Server 2000 Analysis Services with Service Pack 4.
- Microsoft SQL Server 2005 Notification Services or Microsoft SQL Server 2000 Notification Services 2.0 with Service Pack 1.

- Microsoft Windows SharePoint Services 2.0 with Service Pack 2. If FrontPage Server 2002 extensions are installed on IIS, uninstall them before installing Microsoft Windows SharePoint Services 2.0.

The following operating systems can also be used with BZ 2006:

- Windows XP Professional with Service Pack 2.
- Windows 2000 Server with Service Pack 4 or above.
- Windows Server 2003 R2.

Once the various hardware and software requirements are in place to install BZ 2006, you can go ahead with the installation process, as discussed in the next section.

Installation of BizTalk Server 2006

After analyzing the hardware and software requirements for installing BZ 2006, Let's choose the edition of BZ 2006. It has editions such as:

- **Enterprise Edition:** It is required by customers with enterprise level requirements.
- **Standard Edition:** It is required by customers with moderate business volume requirements
- **Branch Edition:** It is required for performing the RFID requirements.
- **Developer Edition:** It is required by developers for development and testing purpose.

You can download the BizTalk Server 2006 from the Microsoft website and burn the same on the CD for installation. let's go ahead and install the server on your system by the following steps:

1. *Insert* the CD/DVD of BZ 2006. An auto-run menu will appear on your monitor screen, as shown in Fig.Biz-1.7.
2. *Click* the **Install** link from the auto-run menu to start the installation process (Fig.Biz-1.7). A **Customer Information** screen appears, as shown in Fig.Biz-1.8.

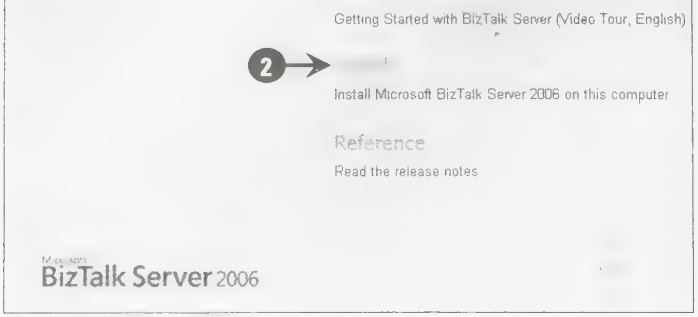


Fig.Biz-1.7

3. In the **Customer Information** screen, *enter* the user name, name of your organization, and product key beside the **User name, Organization**, and **Product key** fields respectively (Fig.Biz-1.8).
4. Now, *click* the **Next** button (Fig.Biz-1.8) to open the **License Agreement** screen, as shown in Fig.Biz-1.9.

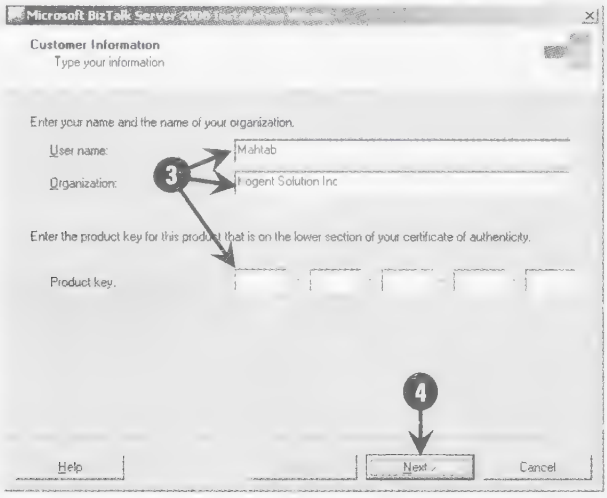


Fig.Biz-1.8

5. After carefully reading the terms of the **License Agreement**, *select* the radio button beside the **Yes, I accept the terms of the license agreement** option (Fig.Biz-1.9).
6. Now, *click* the **Next** button (Fig.Biz-1.9) to open the **Component Installation** screen, as shown in Fig.Biz-1.10.

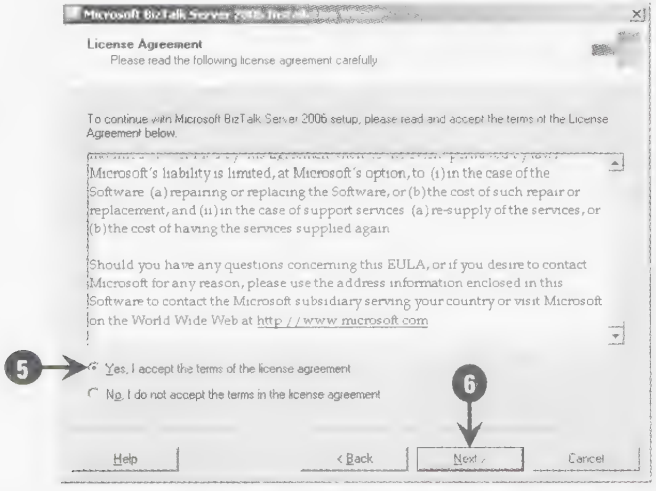


Fig.Biz-1.9

In the **Components Installation** screen, you will find all the components under the **Available component** box already tick marked and selected by default (Fig.Biz-1.10).

7. *Enter* the location where you want to install BZ 2006, beside the **Install to** option (Fig.Biz-1.10).
8. Now, *click* the **Next** button (Fig.Biz-1.10). This opens the **Summary** screen of the **Installation Wizard**, as shown in Fig.Biz-1.11.

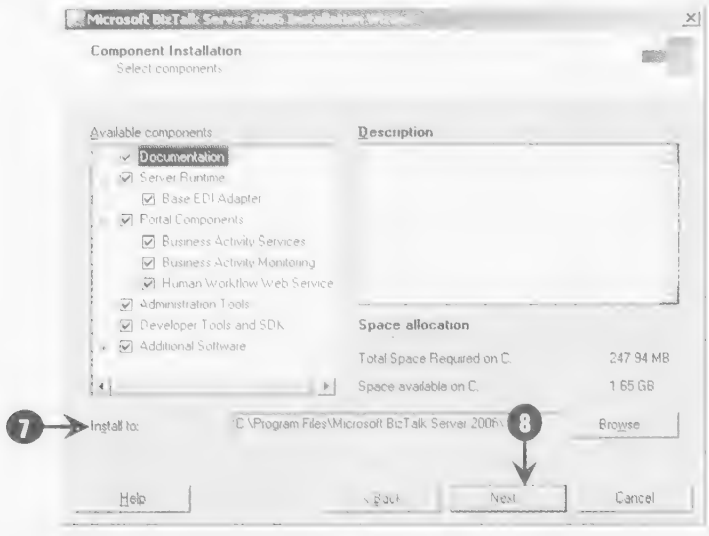


Fig.Biz-1.10

9. The **Summary** screen provides a list of components of BZ 2006 selected by default in the previous step which the **Installation Wizard** will install in your system. *Click* the **Install** button to begin installing the components (Fig.Biz-1.11). You can click on the **Back** button (Fig.Biz-1.11) to go back the **Component Installation** screen and deselect the selected component that you don't need to install.

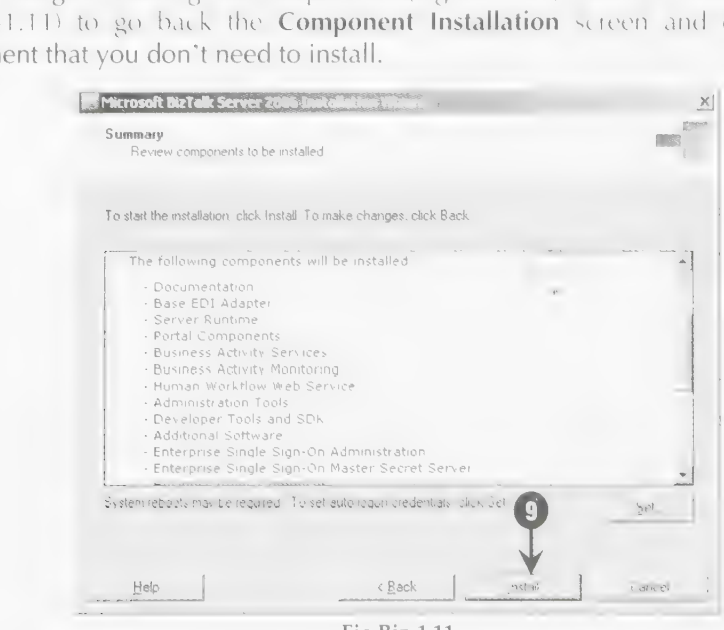


Fig.Biz-1.11

The **Installation Progress** screen appears, as shown in Fig.Biz-1.12. This screen shows the progress of the installation process. The **Next** button (Fig.Biz-1.12) is deactivated. This button will be activated as soon as the installation progress is complete.

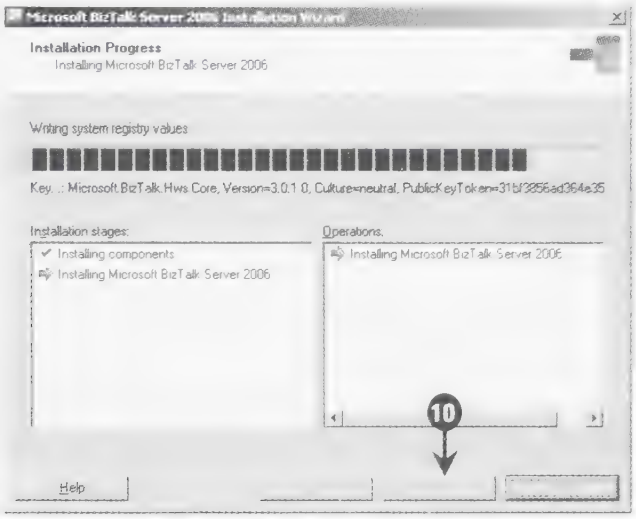


Fig.Biz-1.12

10. *Click* the **Next** button once the installation process is complete (Fig.Biz-1.12). The **Installation Completed** screen appears, as shown in Fig.Biz-1.13.
11. *Click* the **Finish** button in the **Installation Completed** screen (Fig.Biz-1.13). The **Microsoft BizTalk Server 2006 Configuration** screen appears, as shown in Fig.Biz-1.14.

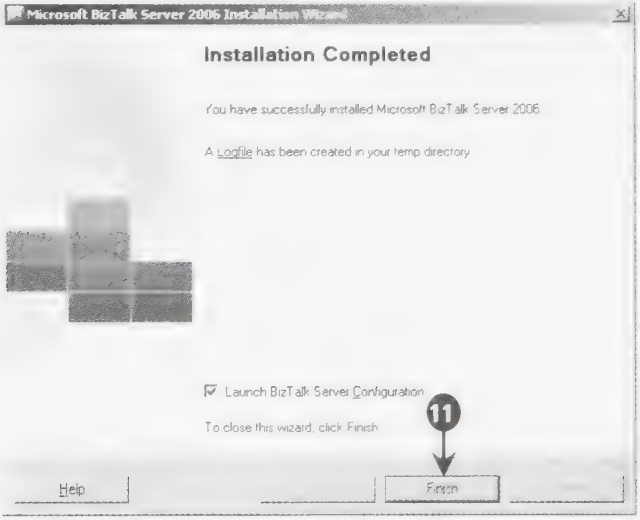


Fig.Biz-1.13

After successfully installing BZ 2006, the next step is to configure the server. This is taken up in the next section.

Configuring BizTalk Server

Once we have installed BZ 2006, we have to configure the server. This can be done by either of the following methods:

- Basic configuration
- Custom configuration

Basic configuration

Basic configuration is used by beginner level developers to configure BZ 2006. To configure BZ 2006 using the basic configuration method, undertake the following steps:

1. In the **Microsoft BizTalk Server 2006 Configuration** screen, the Basic configuration radio button is selected by default, along with the name of SQL server beside the **Database server name** option. In this screen, under **Service credential** type the username and password beside the **User name** and **Password** text boxes respectively, as shown in Fig.Biz-1.14.
2. Next, *click* the **Configure** button (Fig.Biz-1.14).

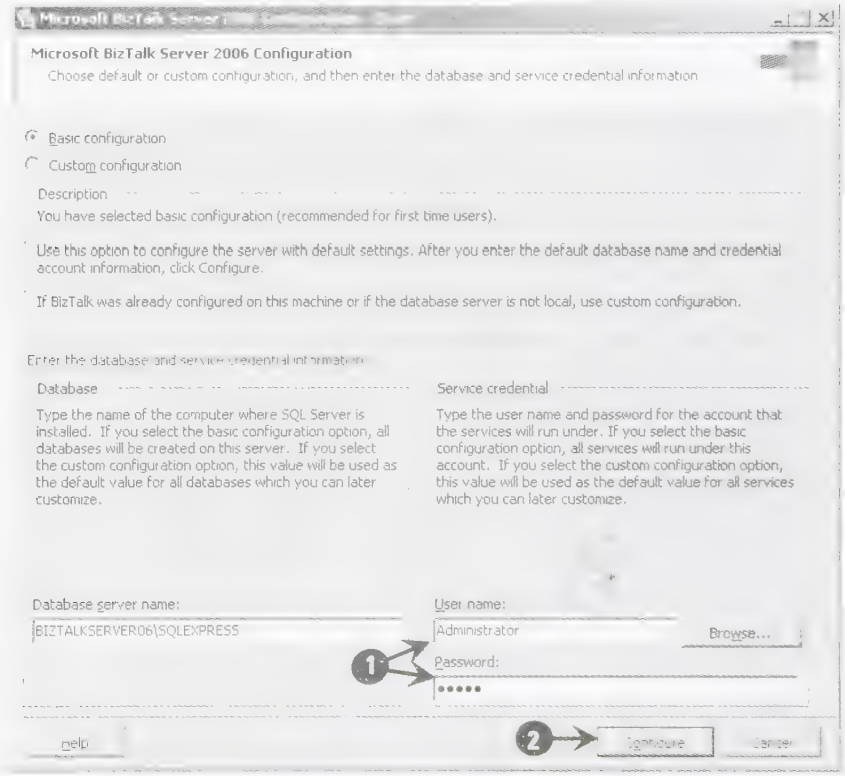


Fig.Biz-1.14

3. The **Summary** screen appears with a list of components of BZ 2006 that you want to configure. *Click* the **Configure** button to automatically configure these components, as shown in Fig.Biz-1.15.

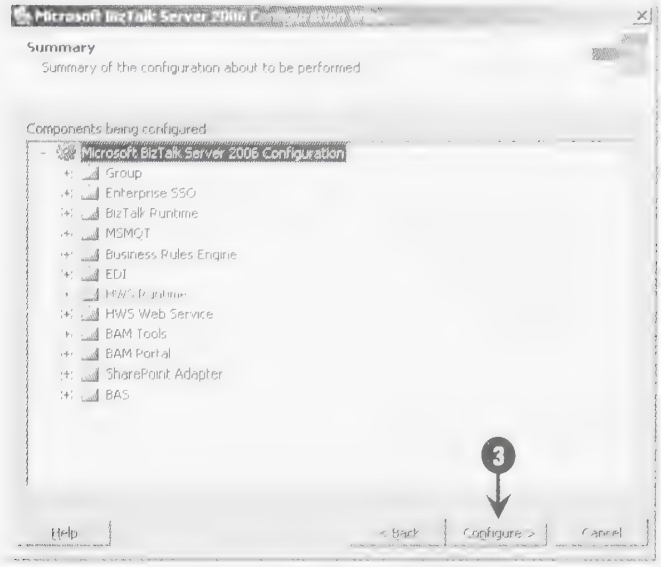


Fig.Biz-1.15

The **Configuration Progress** screen appears. Here, the **Configuration Wizard** automatically configures all necessary components, such as **Group**, **Enterprise SSO**, and **BizTalk Runtime**. A green tick mark indicates the successful completion of a component configuration while a red cross indicates failure, as shown in Fig.Biz-1.16.

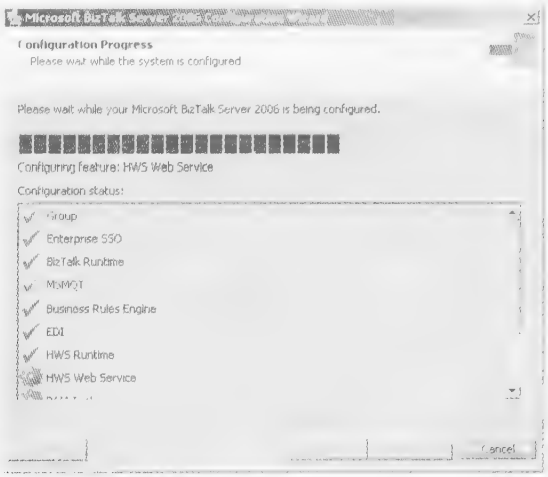


Fig.Biz-1.16

When the configuration process is complete, a screen with a list of the components appears, as shown in Fig.Biz-1.17. In the figure, you can check the configuration result of all the components. The components that are successfully configured will have 'Success' against them under **Configuration Result**, while those that could not be configured will have 'Failure' marked against them. You can select the **Launch Manual Configuration application** option to manually configure the failed components (Fig.Biz-1.17).

4. **Next**, *click* the **Finish** button to close the Basic Configuration wizard (Fig.Biz-1.17).

Fig.Biz-1.17

You now know how to configure the various components of BZ 2006 according to the Basic configuration method. If any of the components is not configured successfully through this method, these components can be configured using the **Custom** configuration method.

Custom configuration

The Custom configuration method allows you to configure BZ 2006 in advanced configuration mode. In this configuration method, you can configure or unconfigure each component manually. To configure the various components of BizTalk Sever using the Custom configuration method, follow these steps:

1. *Select* the radio button beside the **Custom configuration** option in the **Microsoft BizTalk Server 2006 Configuration** screen (Fig.Biz-1.14).
2. Next, *click* the **Configure** button. A screen similar to the one shown in Fig.Biz-1.18 appears.

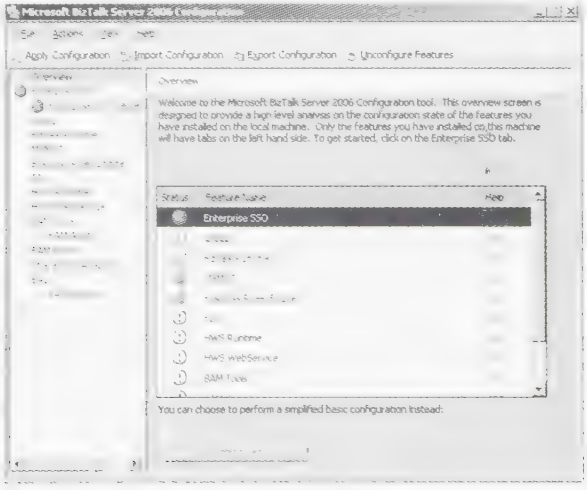


Fig.Biz-1.18

You can use the screen shown in Fig.Biz-1.18 to configure the various components of BZ 2006 manually. Now, you must be thinking about enabling enterprise SSO which helps in enabling Single Sign-On service for all BZ service.

We will continue with the chapter and describe how to install the Enterprise Single Sign-On (SSO) component of BZ 2006 in the next section.

Installing Enterprise Single Sign-on

The Enterprise Single Sign-on (SSO) a component enables a user to authenticate the resources of multiple software systems in BZ 2006 with a single login. It provides capabilities for securely storing and transmitting user credentials across local and network boundaries. Enterprise SSO can be installed by the following steps:

1. *Run* the BZ 2006 setup and then perform steps 1 to 7 of the 'Installation of BizTalk Server' section of this chapter. The **Component Installation** screen appears, as shown in Fig.Biz-1.19.

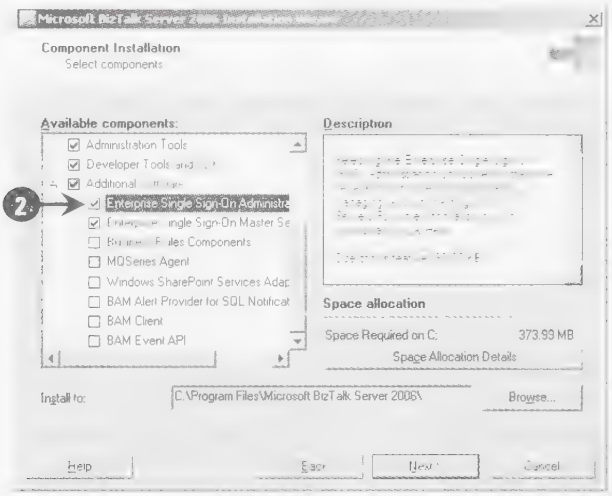


Fig.Biz-1.19

2. In the Component Installation screen, expand the Additional software node and tick mark the **Enterprise Single Sign-On Administration** and **Enterprise Single Sign-On Master Secret Server** options to install Enterprise SSO (Fig.Biz-1.19). These options are briefly described:
   - **Enterprise Single Sign-On Administration:** This option is used when you want to map and connect the Enterprise Single Sign-On Services in administration and client tools. It is used to create and manage applications and perform all the administration tasks that application administrators and users need through BZ 2006.
   - **Enterprise Single Sign-On Master Secret Server:** This option allows you to create the SSO database and acts as a Master Secret Server in the Single Sign-On System. The Master Secret Server is used to store the master secret (encryption key). This encryption key is used to encrypt and decrypt the data it stores in the SSO database.
3. Now, perform steps 8-11 of the 'Installation of BizTalk Server' section of this chapter.

Once you have successfully installed the Enterprise Single Sign-On in your system, the next step is to run SSO by the following step:

4. *Click* **Start→Programs→Microsoft Enterprise Single Sign-On→SSO Administration**.

Next, we will learn to install another SSO component, called the SSO Client utility.

Installation of SSO Client Utility

The SSO Client utility allows end users to configure their client information in the SSO database. A self-extracting **SSOClientInstall.exe** file is used to install SSO Client Utility. To install SSO Client Utility, you must have one of the following:

- Microsoft Windows Server 2003.
- Microsoft Windows 2000 Server with Service Pack 4, or Windows XP Professional with Service Pack 1.
- .NET Framework 2.0. This version is necessary only if you are using the UI-based SSO Client Utility.

Now, to install the SSO Client utility, follow the steps given here:

1. *Double click* the self-extractor **SSOClientInstall.exe** file in your BZ 2006 installation CD. The WinZip Self-Extractor program appears on your screen.
2. Then, *select* the folder where you want to unzip these files and *click* the Unzip button. The **Microsoft Enterprise Single Sign-On Client Setup** screen appears, as shown in Fig.Biz-1.20.

Fig.Biz-1.20

3. In this screen *click* the **Next** button (Fig.Biz-1.20). The **User Information** screen appears, as shown in Fig.Biz-1.21.

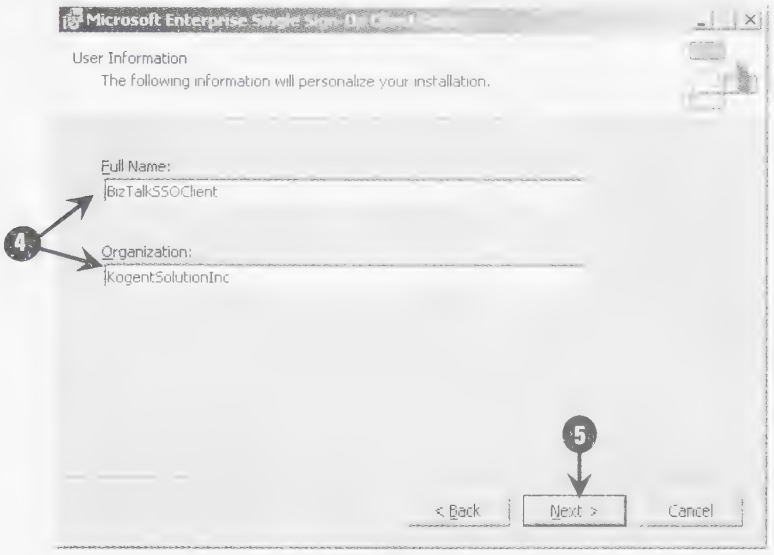


Fig.Biz-1.21

4. *Enter* your name and the name of your organization under the **Full Name** and **Organization** text boxes respectively (Fig.Biz1.21).
5. Now, *click* the **Next** button (Fig.Biz-21). The **Start Installation** screen appears, as shown in Fig.Biz-1.22

Fig.Biz-1.22

6. *Click* the **Install** button to begin the process of installing the SSO Client utility (Fig.Biz-1.22). Once the installation process is over, the **Completing the Microsoft Enterprise Single Sign-On Client Wizard** screen appears, as shown in Fig.Biz-1.23.

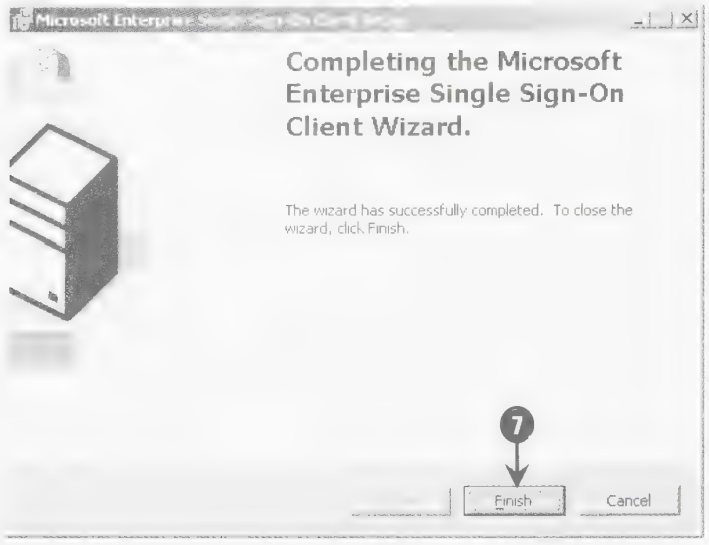


Fig.Biz-1.23

7. *Click* the **Finish** button to close the SSO Client utility setup (Fig.Biz-1.23).

After successful installation of the SSO Client utility, *Click* **Start→Programs→Microsoft Enterprise Single Sign-On→SSO Client Utility** to run the SSO Client Utility.

Next, we will describe the various tools available in BizTalk Server 2006. These tools are helpful when you build the BizTalk application.

Tools in BizTalk Server 2006

There are various types of tools available in BZ 2006. Some of the important tools are

- BizTalk Schema Editor tool
- Orchestration tool
- Mapper tool
- Pipeline tools
- BizTalk Explorer tool

BizTalk Schema Editor Tool

The BizTalk Schema Editor tool runs within the Microsoft Visual Studio 2005 environment. You can use this tool to create, edit and manage schemas within your application. BizTalk Schema Editor uses its own graphical system to represent the structure of messages, and uses the XML Schema Definition (XSD) language to store instances of messages and to exchange messages in specific format. For example, suppose you want to exchange flat files. To do this, BizTalk Server 2006 processes the flat files and converts them to XML format. One of the most important reasons for translating all instance messages into XML format is to simplify the process of transforming a message from one structure into another.

Orchestration Tool

An orchestration is a powerful and flexible tool used for representing business processes based on **XLANG/s** language. **XLANG/s** is a messaging language with some expression capabilities of **C#. XLANG/s** language is executed by orchestrations that perform business processes. Messages are sent and received by orchestrations through **MessageBox**, as shown in Fig.Biz-1.24. To know more about orchestrations, please refer to Chapter 2. An orchestration can be developed under Visual Studio 2005, which will be discussed in section 'Adding an Orchestration' of Chapter 3.

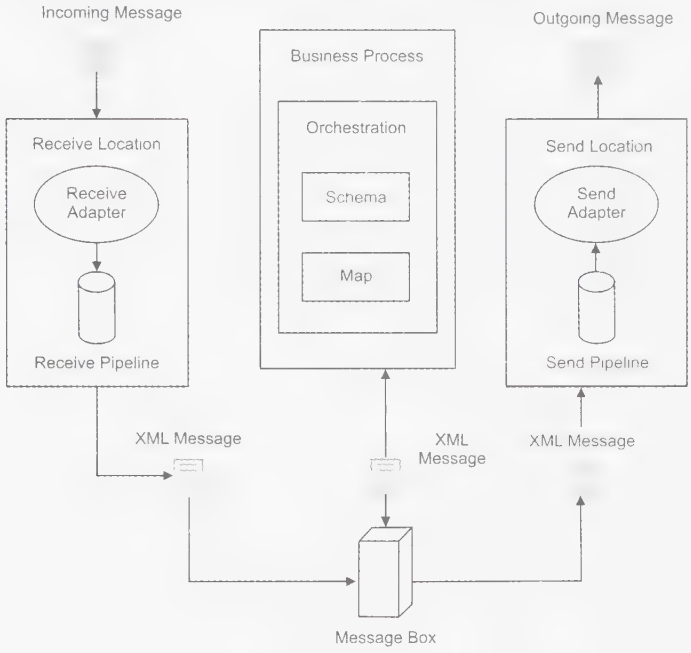


Fig.Biz-1.24

Mapper Tool

The BizTalk Mapper tool is used to establish the relationship between an input and an output schema by using linking and functoids. A linking is when a data of a record or field is directly connected to items in another schema. Functoids mainly manipulate complex data, such as adding the value of two fields in the source schema, copying the result to the destination schema, converting a character to ASCII format and returning the average of a field in a repeating record. BizTalk Mapper stores maps in a file with a .btm extension. This file saves design information about the map, the locations of icons representing functoids, and the linking between schema items and functoids.

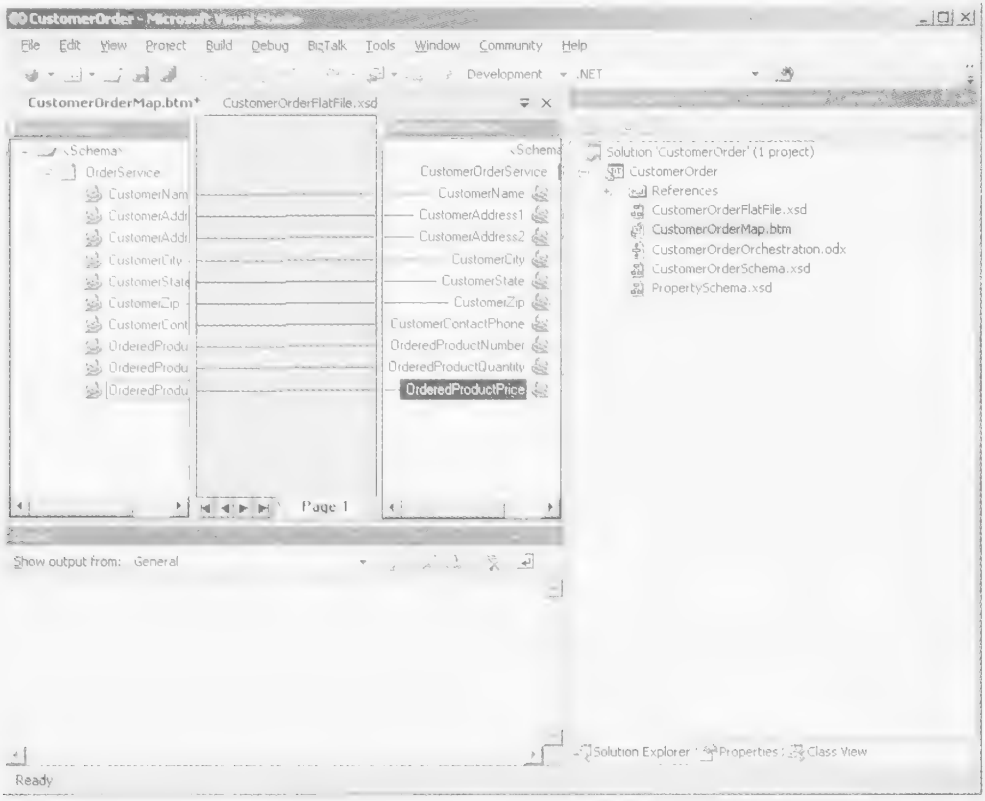


Fig.Biz-1.25

Fig.Biz-1.25 shows how each field of source schema can be mapped to corresponding fields of the destination schema. It also shows how to translate and transform messages by converting them from one format to another format, such as converting a flat file into an XML file. To know more about the Mapper tool, please refer to the 'Mapping the Message' section of Chapter 4.

Pipeline Tools

The pipeline tools (which include the Send and Receive pipelines) transform XML messages from source nodes to destination nodes and perform other data-specific actions, such as data encryption and decryption. The main tasks of the pipeline tools are as follows:

- They transform data of various formats to XML format and vice-versa.
- They decode and encode documents.
- They encrypt and decrypt documents.
- They sign the digital signature of documents for verification.

Fig.Biz-1.24 shows the workflow of the pipelines involved in processing messages. The incoming messages first pass through the Receive Adapter. From there they go to the Receive Pipeline, where they are transformed to XML messages. After these XML messages are transmitted from the Receive Pipeline, they can be used by orchestrations and then sent to the Send Pipeline for the destination node as outgoing messages.

BizTalk Explorer Tool

The BizTalk Explorer tool is a Microsoft Visual Studio 2005 tool that displays the contents of BizTalk Management database. It enables you to explore, configure, manage and reuse orchestrations as well as receive locations and send ports through a graphical interface in the Visual Studio 2005 development environment. The main tasks of the BizTalk Explorer tool are as follows:

- It enables you to view and manage the configuration details of your project.
- It is used in viewing databases and assemblies.
- It is used to make and edit ports, roles, and parties.
- It is used to deploy/undeploy business processes.

BizTalk Management database is the central storage information used to store artifacts, such as orchestrations, receive locations, send ports, party, and so on. For more information on BizTalk Explorer tool, please refer to the 'Using the BizTalk Explorer' section of Chapter 4.

With this, we come to the end of the chapter. We hope the chapter has served the purpose of familiarizing you with the basic features of BZ 2006. Next is a short summary of the chapter.

Summary

The chapter described the latest version of BizTalk Server, known as Microsoft BizTalk Server 2006 along with its features. It dealt with the basic concepts and types of architecture of BZ 2006, such as simple, runtime, and BAS architecture. Also discussed were the installation and configuration processes of BZ 2006 in simple steps, after which the chapter explained in depth the various tools available in the BZ 2006 server.

In the next chapter, we will discuss the business process and orchestrations designer components that will help you to build the BizTalk application.

Chapter 2

Exploring Business Process

In this Chapter

- Building a Business Process
- Understanding BizTalk Orchestration Designer Surface

A business process is a set of coordinated tasks and activities, conducted by both people and equipments, which lead to accomplishing a specific organizational goal. Business process management (BPM) is a systematic approach to improving these processes. BizTalk is a business process management (BPM) server that enables companies to automate and optimize business processes by exchanging business documents (for example, purchase orders and invoices) between business applications within or across organizational boundaries.

You can easily build and implement a business process through BizTalk Server 2006. For this, you have to add various BizTalk components, such as orchestration, messaging, ports and orchestration designer shapes. BizTalk orchestration is the brain of the BizTalk application. It enables you to design and implement business processes by simply dragging and dropping various orchestrations shapes available in the Orchestration designer tool. This chapter introduces you to the various components that are involved in the making of a business process.

Building a Business Process

As already said, a business process is a collection of activities designed to produce a desired goal in an organization. It implies a strong emphasis on how the work is done within the organization. A business process is thus a specific ordering of work activities across time and place, with a beginning, an end, and clearly defined inputs and outputs

BizTalk Server 2006 builds a business process in a set of logically sequenced operations. To build a business process, you need to model the process by creating a business process orchestration using the orchestration designer shapes in BizTalk. A business process orchestration is the graphical representation of a business process. You can create a business process orchestration by using different shapes available in Orchestration designer shapes Toolbox. After creating a business process orchestration, the next task is to add ports, messages, and orchestration designer shapes in the BizTalk project by using Microsoft Visual Studio 2005. All BizTalk applications are developed under the Microsoft Visual Studio 2005 environment.

Now that we know the meaning of a business process as well as the various components needed to implement the process through BizTalk Server 2006, we can go ahead and learn about the various parts of the BizTalk orchestration designer, which we are going to use to add various shapes required to build the business in our project. After this, we take you through the steps to develop a simple business process in a BizTalk project.

Understanding BizTalk Orchestration Designer Surface

The orchestration designer surface is a graphical add-in that a developer can use to develop a BizTalk project. It is used for developing business processes of an organization and is hosted with Visual Studio 2005 to enable developers create a BizTalk orchestration. It also helps the business analysts model the processes, configure the messaging operations required in these processes and implement them. You can use the orchestration designer to create XLANG schedule drawings. These drawings are compiled to an XML-code schedule that can be run by an XLANG scheduler engine. XLANG is an XML business process language used for orchestrating applications that coordinate the workflow among the various components of the BizTalk project as well as to help build business processes in BizTalk Server.

The BizTalk orchestration designer is shown in Fig.Biz-2.1. As can be seen from the figure, the orchestration designer is divided into the following parts:

- **Orchestration Surface**: This is at the center of Visual Studio screen. It is here that shapes are dropped and the logic of the business process modeled. It is surrounded on either side by two port surfaces.
- **Port Surface**: The two port surfaces are used to drop port shapes to define the interface between orchestrations and messaging. Messages enter the orchestration through these ports. To know more about port shapes, please refer to the 'Port shape' section later in this chapter.
- **Output pane**: At the bottom of the orchestration designer is the Output pane, which is used to view error and warning messages generated during the building of the BizTalk project.
- **Orchestration view**: This view is in a tree-like structure whose nodes represent all the non-visual elements in an orchestration. These include messages, ports, variables (used to write BizTalk expressions), parameters (parameters are just like those that are passed in a function of any programming language. In BizTalk Server, you can specify the parameters of your orchestration, such as messages, variables, and ports) and port types declared in the orchestration. A developer can add and configure these components in this view during the development of the BizTalk project.
- **Properties pane**: This pane is used to set or change the names and other attributes of BizTalk components such as orchestration, ports and messages.
- **Orchestration designer shapes (Toolbox)**: This Toolbox is visible on the left of the Microsoft Visual Studio 2005 environment. The components visible in the Toolbox include shapes used in constructing business process orchestrations. To use these shapes, simply drag them from the Toolbox onto the orchestration surface area.

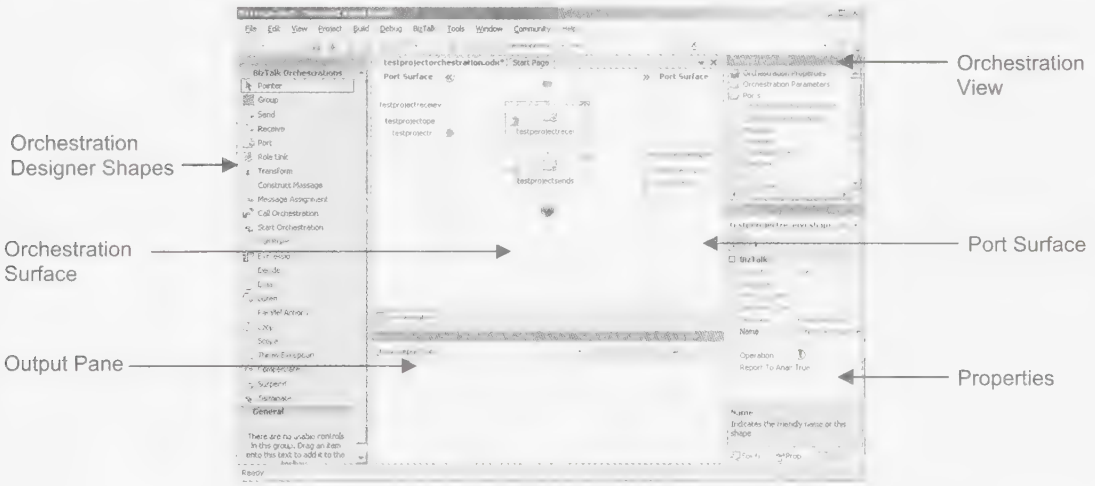


Fig.Biz-2.1

You are now familiar with the various parts of the BizTalk orchestration designer. You can use this knowledge to develop a simple business process by using the various orchestration designer shapes. However, to do this, we need to first create a BizTalk project. Let's learn how to do this next.

Creating a BizTalk Business Process

A BizTalk project contains different interrelated tasks of a business process required to run an organization. A business process is built by using the various components of the orchestration designer. However, before we can build business processes as well as add the various components needed to run them, we need to create a BizTalk project. A BizTalk project can be created by following these steps:

1. *Click* **Start→All Programs→Microsoft Visual Studio 2005→Microsoft Visual Studio 2005** to open the **Microsoft Visual Studio** screen, as shown in Fig.Biz-2.2.

Fig.Biz-2.2

2. *Select* **File→New→Project** to open the **New Project** dialog box, as shown in Fig.Biz-2.3.

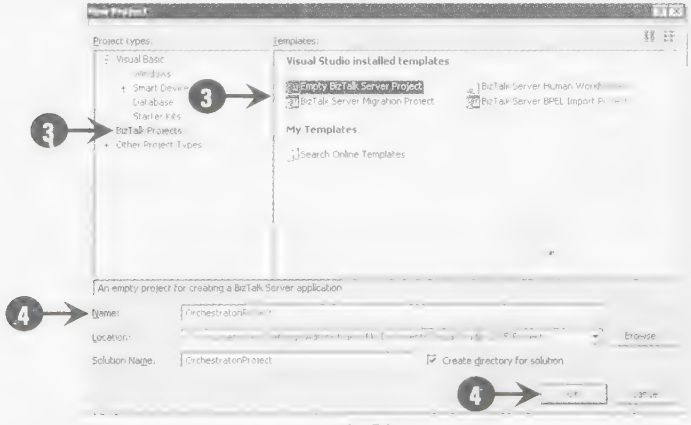


Fig.Biz-2.3

3. *Select* **BizTalk Projects** and **Empty BizTalk Server Project** from the **Project types** and **Templates** panes respectively (Fig.Biz-2.3)
4. *Enter* the name of the BizTalk project as **OrchestratonProject** beside the **Name** text box and *click* the **OK** button to create a BizTalk project, as shown in Fig.Biz-2.4.

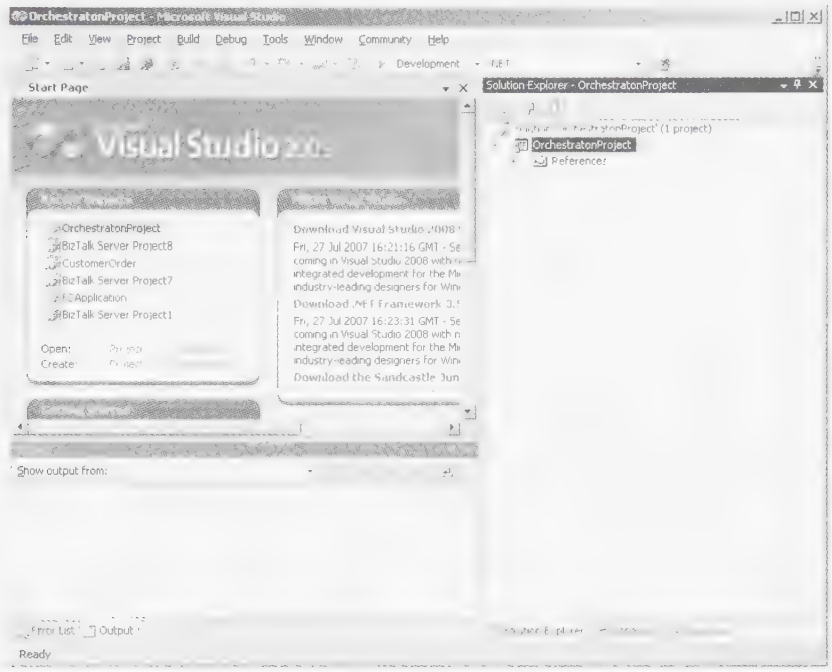


Fig.Biz-2.4

5. *Right-click* **OrchestrationProject** from Solution Explorer and select **Add→New Item** from the context menu to add an orchestration file. The **Add New Item–OrchestrationProject** dialog box appears, as shown in Fig.Biz-2.5.
6. *Enter* the name of orchestration as **Chapter2Orchestration.odx** beside the **Name** text box (Fig.Biz-2.5).

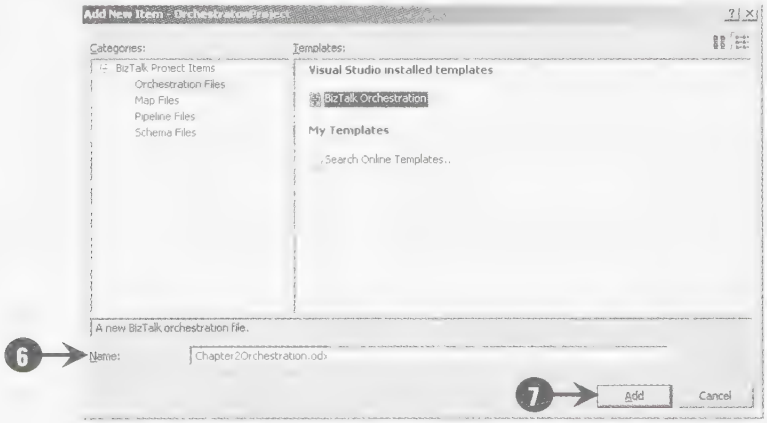


Fig.Biz-2.5

7. Now, *click* the **Add** button (Fig.Biz-2.5) to add the orchestration in the **OrchestrationProject** project, as shown in Fig.Biz-2.6.

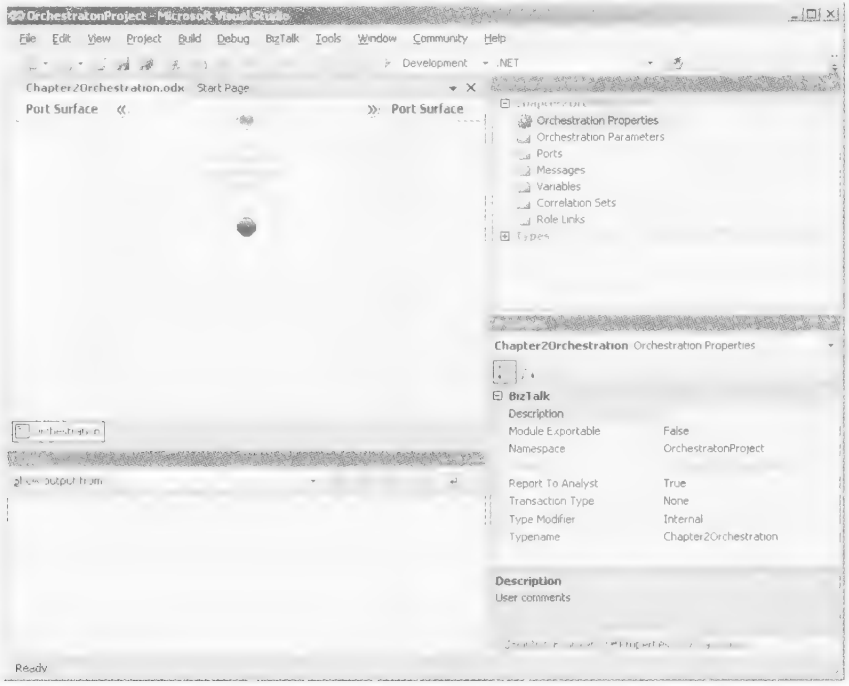


Fig.Biz-2.6

Thus, the new orchestration, **Chapter2Orchestration.odx**, is created in **OrchestrationProject**, a shown in Fig.Biz-2.6. To add shapes in the **Chapter2Orchestration.odx**. orchestration, simpl drag the shapes from the Toolbox and drop them in the area labeled 'Drop a shape from th toolbox here' (Fig.Biz-2.6). To know more about orchestration, please refer to the 'Tools i BizTalk Server 2006' section of Chapter 1 and the 'Adding an Orchestration' section c Chapter 3.

Next, we will describe the various types of shapes available in orchestration designer (Toolbox that are used to build business processes.

Introducing Orchestration Designer Shapes

The orchestration designer shapes are the basic building blocks through which orchestration implement business processes. Each shape has a specific visual appearance representing specific task to build the business process.

A business process is a set of actions that together meet some useful business need. A develope can use the orchestration designer in the BizTalk Server 2006 engine to define these action graphically. Rather than expressing the actions in a programming language, the orchestratio designer allows a developer to create an orchestration by simply connecting together a series c shapes in a logical way.

When you add these shapes onto the orchestration surface, they connect with each other t build the business process. The left side of the Microsoft Visual Studio 2005 screen contains th **Toolbox** pane, as shown in Fig.Biz-27. This pane displays various orchestration designer shape that are used to build business process. All you have to do is simply drag these shapes from th

Toolbox and drop them in the orchestration surface. Fig.Biz-2.7 shows the Receive and Send shapes added in the orchestration surface.

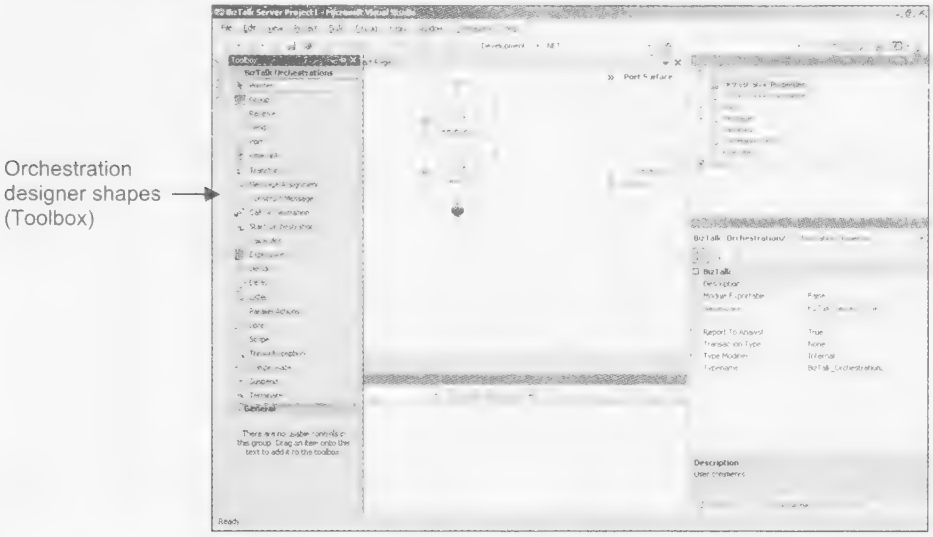


Fig.Biz-2.7

You can use the orchestration designer shapes to design a business process orchestration. These shapes represent the operations required to be performed in the business process. The shapes available in Toolbox of Microsoft Visual Studio 2005 may be categorized as follows:

- Transferring control shapes, used in transferring the control from one shape to another. These include the **Decide** and **Loop** shapes.
- Messaging shapes, used in passing messages between applications in BizTalk Server. These include the **Send** and **Receive** shapes.
- Data shapes, used to construct or transform messages or alter the data within the orchestration. These include the **Transform**, **Construct Message**, and **Message Assignment** shapes.
- Error handling shapes, used to throw an exception or handle errors in the business process orchestration. These include the **Throw Exception** and **Suspend** shapes.

In all, these broad categories include the following shapes:

- The Send Shape
- The Receive Shape
- The Port Shape
- The Call orchestration Shape
- The Start orchestration Shape
- The Loop Shape
- The Decide Shape
- The Construct Message Shape

- The Transform Shape
- The Message Assignment Shape

Let's now discuss each of these shapes in detail.

The Send Shape

The Send shape enables you to send messages from a port in a business process orchestration. It also transmits the message to the BizTalk MessageBox and returns immediately to the next task of the business process. In BizTalk Server, you need to specify a message and port operation for a **Send** shape. The port operation helps to send messages to the **Send** shape. Fig.Biz-2.8 displays the **Send** shape.

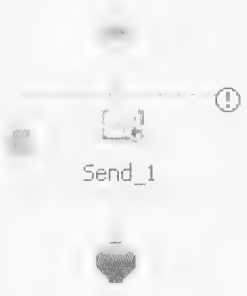


Fig.Biz-2.8

As you can see in the figure, the **Send** shape contains a circle with a red exclamation mark smart tag, which helps to configure the **Send** shape. For more information on how to configure the **Send** shape, please refer to the 'Adding receive/send shape' section of Chapter 3.

The Receive Shape

The **Receive** shape enables you to receive a message from a port in a business process orchestration. Once a **Receive** shape has been dragged onto the orchestration surface, it must be configured by the exclamation mark smart tag. Fig.Biz-2.9 shows the **Receive** shape.

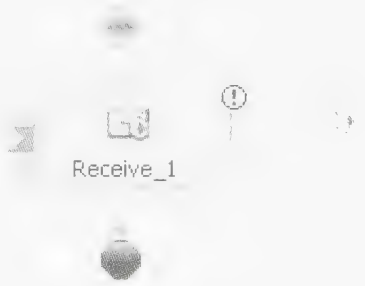


Fig.Biz-2.9

The figure shows the Receive_1 Receive shape that you need to connect to a port for receiving the message of a specific type such as XML, BizTalk schema and flat-file schema message. For more information on how to configure and specify a message type onto the Receive shape, please refer to the 'Adding receive/send shapes' section of Chapter 3.

The Port Shape

The **Port** shape acts as a logical representation of the input/output location for a message to interface with an orchestration. The Port shape receives or sends messages in a business process orchestration. Each Port shape has a port type that defines which message is to be sent or received, the mode of communication pattern, such as one-way or request-response, and port binding. In port binding, you have to specify the direction of the communication from one location to another. You have to specify the direction of the communication of the message during the configuration of the Port shape by selecting either **I'll always be receiving messages on this port** or **I'll always be sending messages on this port** from the **port direction of communication** drop-down menu. Fig.Biz-2.10 shows the Port shape.

Fig.Biz-2.10

In Fig.Biz-2.10, the name of the port is **Port_1**, and **Operation_1** is a set of operations to handle request or response messages, and the green arrow is used to connect with the Send or Receive shapes. For more information on how to add and configure the Port shape, please refer to the 'Adding Ports in the Project' section of Chapter 3.

The Call Orchestration Shape

The Call Orchestration shape enables you to synchronously invoke other orchestrations of the business process. In a synchronous call, the Call Orchestration shape calls another business process orchestration only after first completing the ongoing business process orchestration. Fig.Biz-2.11 shows the Call Orchestration shape.

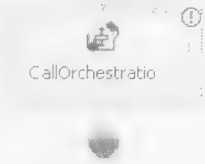


Fig.Biz-2.11

The figure shows the Call Orchestration shape (**CallOrchestration_1**) that is used to call another orchestration. You can call other orchestrations within a BizTalk orchestration without sending a message outside an orchestration's context. The Call Orchestration shape allows a parent orchestration (the main orchestration) to call a child orchestration (the orchestration inherited from the main orchestration) synchronously. Let's suppose that you want to synchronously call an orchestration. To do this, you have to add the **Calling Orchestration** shape by following these steps:

1. *Drag* a **Call Orchestration** shape from the **Toolbox** onto your orchestration surface, as shown encircled in Fig.Biz-2.12.

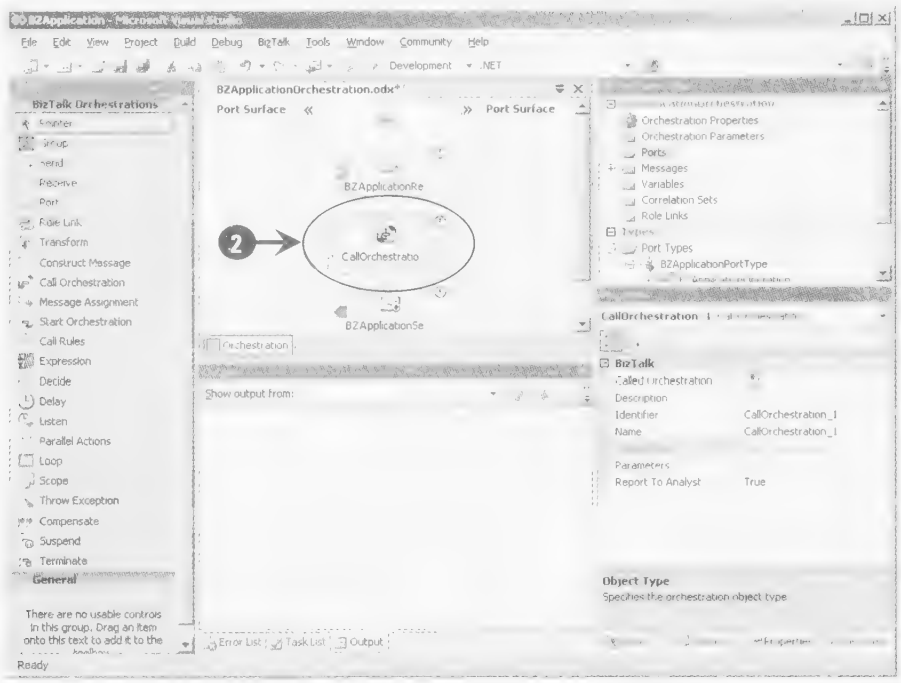


Fig.Biz-2.12

2. *Click* the exclamation mark on the **CallOrchestration_1** shape and then select **No Called Orchestration. Click to configure** option. The **Call Orchestration Configuration** dialog box appears, as shown in Fig.Biz-2.13

3. *Click* the down arrow beside the **Select the orchestration you wish to call** option and select **BZApplication.BizTalk_Orchestration1** from the drop-down list of the **Call Orchestration Configuration** dialog box (Fig.Biz-2.13).

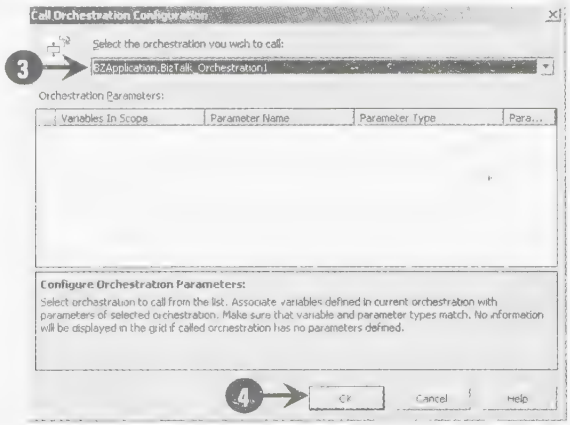


Fig.Biz-2.13

4. *Click* the **OK** button to close the **Call Orchestration Configuration** dialog box and return to your orchestration.

Thus, the new orchestration, **Orchestration_1**, is called with the **BZApplication.BizTalk_Orchestration1** Called Orchestration. When we call another orchestration within BizTalk project, this orchestration must be available in your BizTalk project. It is necessary to first add the orchestration before calling it through the Call Orchestration shape.

The Start Orchestration Shape

The Start Orchestration shape asynchronously invokes other business process orchestrations of a business process. In an asynchronous call, the Start Orchestration shape calls another business process orchestration without waiting for the ongoing business process to be completed. When configuring the Start Orchestration shape, you must first specify the orchestration that you want to invoke by either double-clicking the Start Orchestration shape or by selecting the target orchestration via the **Called Orchestration** property from the **Properties** pane. Fig.Biz-2.14 shows the Start Orchestration shape.

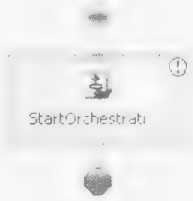


Fig.Biz-2.14

This figure shows the **StartOrchestration_1** shape that is used to start another business process orchestration asynchronously.

The Loop Shape

The Loop shape is the simplest flow control shape that applies a while loop for performing an operation in a business process repeatedly. Each Loop shape executes the task repeatedly until the while condition specified on the Loop shape is false. You can specify a while loop condition by using the BizTalk Expression Editor. Fig.Biz-2.15 shows the Loop shape.

BizTalk Expression Editor is a standard Visual Studio text editor, which you can use to write expressions to expand the capabilities of various orchestration designer shapes. These expressions are similar to those of any programming language. To know more about BizTalk Expression Editor, please refer to the 'Adding the Decide shape' section of Chapter 4.

Fig.Biz2.15

Fig.Biz-2.15 shows the Loop shape that enables you to perform an operation related to a business process repeatedly in a while loop until a condition is false. The condition is in the form of a Boolean expression, which holds the value either true or false.

The Decide Shape

The Decide shape enables you to implement conditional branching in a business process orchestration. It always has at least two branches: a branch for the If statement and a branch for the Else statement. It is equivalent to an If...Then...Else statement in standard programming language. Each branch of a **Decide** shape, except the Else branch, has a rule associated with it. You can use BizTalk Expression Editor to create a Boolean expression in the rule that is evaluated for the execution of that branch. Because the Else branch implies the negation of the Boolean expression in the previous branch, it does not have an expression associated with it. Fig.Biz-2.16 shows the Decide shape.

Fig.Biz-2.16

To know more about how to add the Decide shape, please refer to the 'Adding the Decide Shape' section of Chapter 4.

The Construct Message Shape

The Construct Message shape creates a new message instance in a business process orchestration. It must contain either a Message Assignment shape or a Transform shape but no other shapes. BizTalk messages are immutable, which means that once the messages are created and persisted to the Messagebox database, they cannot be changed at any point. Fig.Biz-2.17 shows the Construct Message shape.

Fig.Biz-2.17

The figure shows the **ConstructMessage_1** shape in which you can drag only the **Message Assignment** shape or the **Transform** shape from the Toolbox and drop them on the **Drop a shape from a toolbox here** location.

The Transform Shape

The Transform shape maps a message to another message in the business process orchestration, thereby transforming the original message. The Transform shape maps one-to-many or many-to-one messages in BizTalk. You have to define one or more input messages, and one or more output messages, and an XSLT (Extensible Stylesheet Language Transformations) map to perform the transformation. XSLT maps are used with XSLT files to map data from one input file (such as XML schema) to other output file (such as flat file schema). Transform shapes are used only in constructing messages, and hence always appear inside a **Construct Message** shape. Fig.Biz-2.18 shows the **Transform** shape.

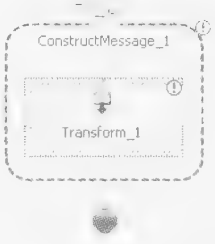


Fig.Biz-2.18

This figure shows the **Transfom_1** shape contained inside the **ConstructMessage_1** shape. To know more about Transform shapes, please refer to the 'Adding a Transform Shape' section of Chapter 4.

The Message Assignment Shape

The Message Assignment shape, as the name suggests, are messages constructed from other messages. It is primarily used to assign an existing message to another message and perform modifications in it, if needed. It also enables messages to be created from .NET classes such as Xml documents. As already mentioned, messages are immutable, meaning that once the messages are assigned, they cannot be changed at any point. Fig.Biz-2.19 shows the **Message Assignment** shape.

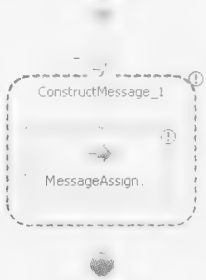


Fig.Biz-2.19

The figure shows the **MessageAssignment_1** shape that allows you to assign one message instance to another message instance.

We hope that you are now familiar with the business process and understand the functions of the various orchestration designer shapes available in BizTalk Server 2006. This also concludes the chapter. A brief summary of this chapter follows next.

Summary

This chapter defined a business process in the context of BizTalk. You also learned about the various surfaces in Visual Studio 2005 with respect to BizTalk Server 2006. Finally, the chapter described the function of the various types of orchestration designer shapes available in BizTalk Server 2006.

In the next chapter, you will learn to develop a simple BizTalk application.

Chapter 3

Creating a Sample BizTalk Applications

In this Chapter

- Selecting the BizTalk Project Type
- Adding an Orchestration
- Message Adding
- Adding Ports in the Project
- Assignment of Strong Naming to the Assembly
- Deploying the Project
- Accessing the Application in BizTalk Server 2006

You must be familiar with business processes and orchestrations by now. To perform a business process, an organization uses custom applications, which cater to the need of the organization's business. These custom applications can be developed by using BizTalk Server 2006 and Microsoft Visual Studio 2005.

In this chapter, we are going to develop a custom sample BizTalk application, which will show the communication of messages from one end to the other in an organization. This custom application will be developed by using Microsoft Visual Studio 2005. For this, you need to first create a BizTalk Project type in the Microsoft Visual Studio 2005. The next step is to add orchestrations, messages, and connecting ports to create a business process. While creating an application, you also need to provide the strong name to the project assembly file for its identification in the global cache. After the application is developed, it can be accessed by configuring it with either BizTalk Server 2006 Administrator Console or BizTalk Explorer.

In this chapter, we will cover and discuss the entire process in simple steps, beginning by learning how to develop a custom sample BizTalk application project.

Selecting the BizTalk Project Type

To develop a BizTalk project, you need to select the BizTalk Project type in Microsoft Visual Studio 2005. BizTalk Project type can be selected by following these steps:

1. *Click* **Start→Programs→Microsoft Visual Studio 2005→Microsoft Visual Studio 2005** to open the **Start Page of** Microsoft Visual Studio window, as shown in Fig.Biz-3.1.

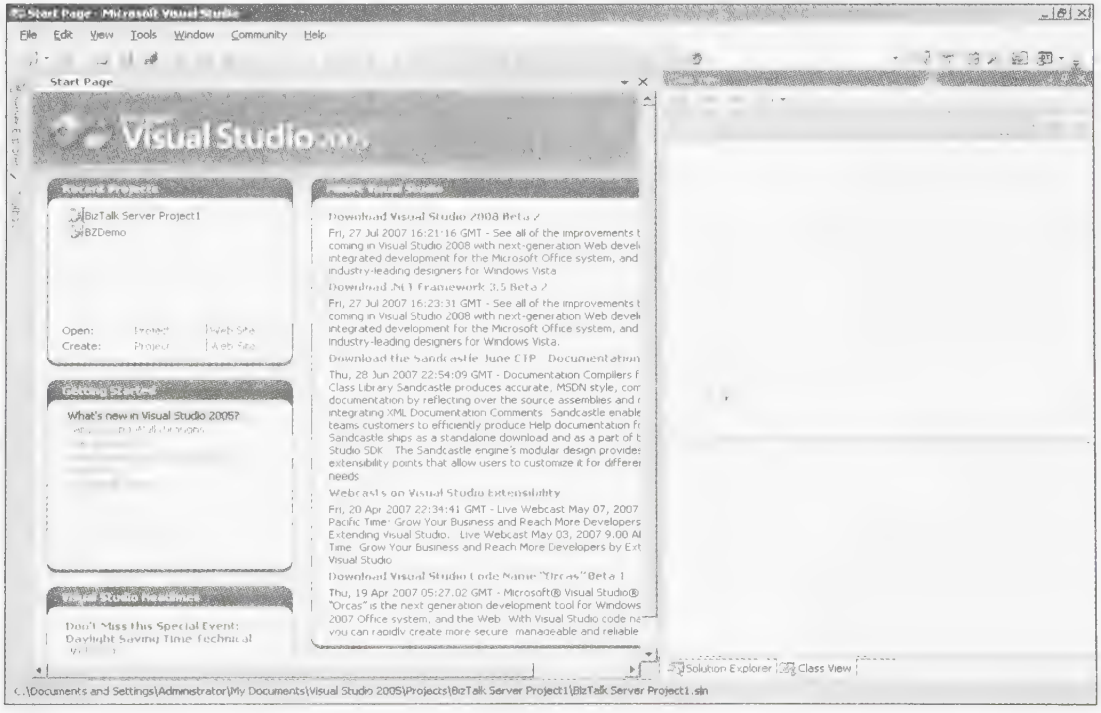


Fig.Biz-3.1

2. *Select* **File→New→Project** on the **Start Page of** Microsoft Visual Studio window to open the **New Project** dialog box, as shown in Fig.Biz-3.2.
3. *Select* **BizTalk Projects** from the **Project types** pane to open the BizTalk templates for creating the BizTalk application, as shown in Fig.Biz-3.2.

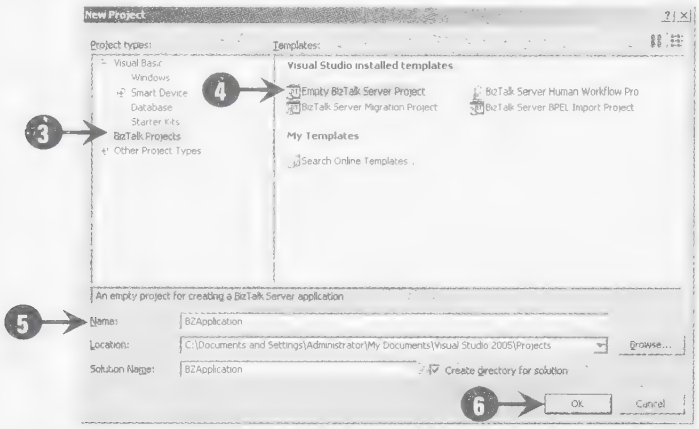


Fig.Biz-3.2

4. *Select* the **Empty BizTalk Server Project** from the **Templates** pane (Fig.Biz-3.2).
5. Next, enter the name of BizTalk project as **BZApplication** in the **Name** textbox.
6. Now, *click* the **OK** button to accept the settings (Fig.Biz-3.2). This opens the **BZApplication - Microsoft Visual Studio** window, as shown in Fig.Biz-3.3.

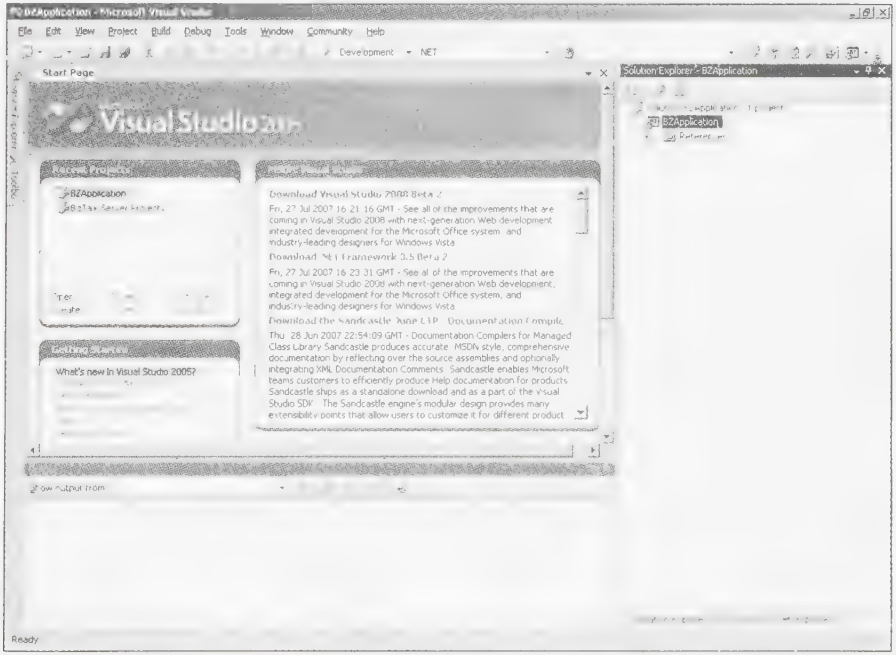


Fig.Biz-3.3

The BizTalk project development environment is now ready. The next step is to add an orchestration in the **BZApplication** project.

Adding an Orchestration

Orchestration is the brain of BizTalk and is used to build business processes for your application. The main feature of orchestrations is that they enable you to write your logic in graphical format rather than in programming language.

Orchestrations can be added in the **BZApplication** project by following these steps:

1. *Right-click* the **BZApplication** project in the **Solution Explorer** (Fig.Biz-3.3) and *select* **Add →New Item** from the context menu to open the **Add New Item – BZApplication** dialog box, as shown in Fig.Biz-3.4.
2. In the **Add New Item – BZApplication** dialog box, *select* the **Orchestration Files** node from the **Categories** pane and select **BizTalk Orchestration** from the **Templates** pane (Fig.Biz-3.4).
3. *Enter* the name of the orchestration as **BZApplicationOrchestration.odx** in the **Name** text box, (Fig.Biz-3.4).

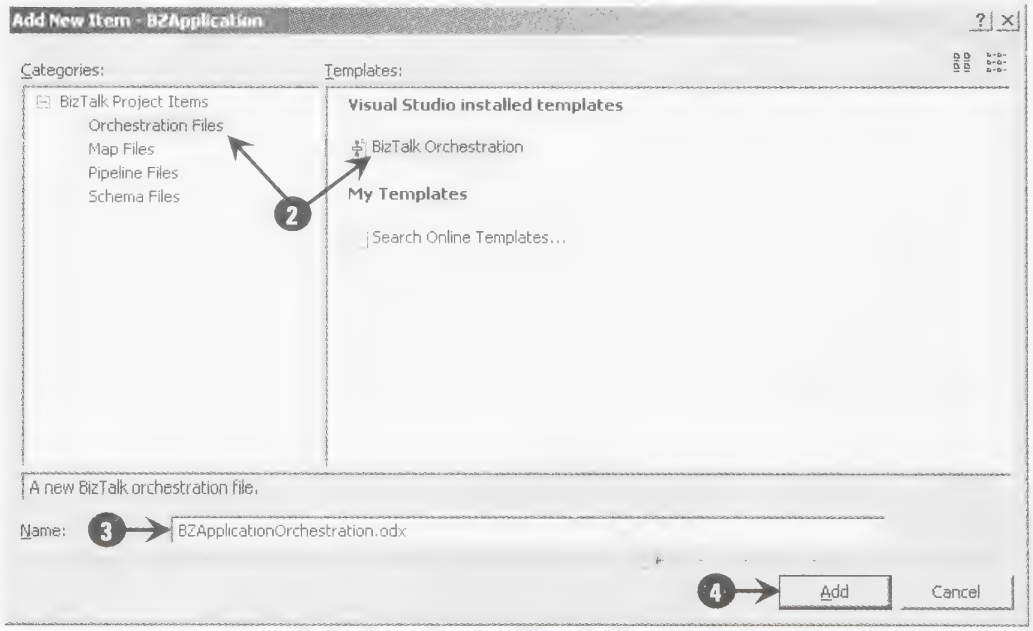


Fig.Biz-3.4

4. Now, *click* the **Add** button to add the **BZApplicationOrchestration** orchestration item to the **BZApplication** project (Fig.Biz-3.4). *Clicking* the **Add** button also opens the **BZApplication - Microsoft Visual Studio** screen, as shown in Fig.Biz-3.5. **BZApplicationOrchestration** is thus added to your **BZApplication** project.

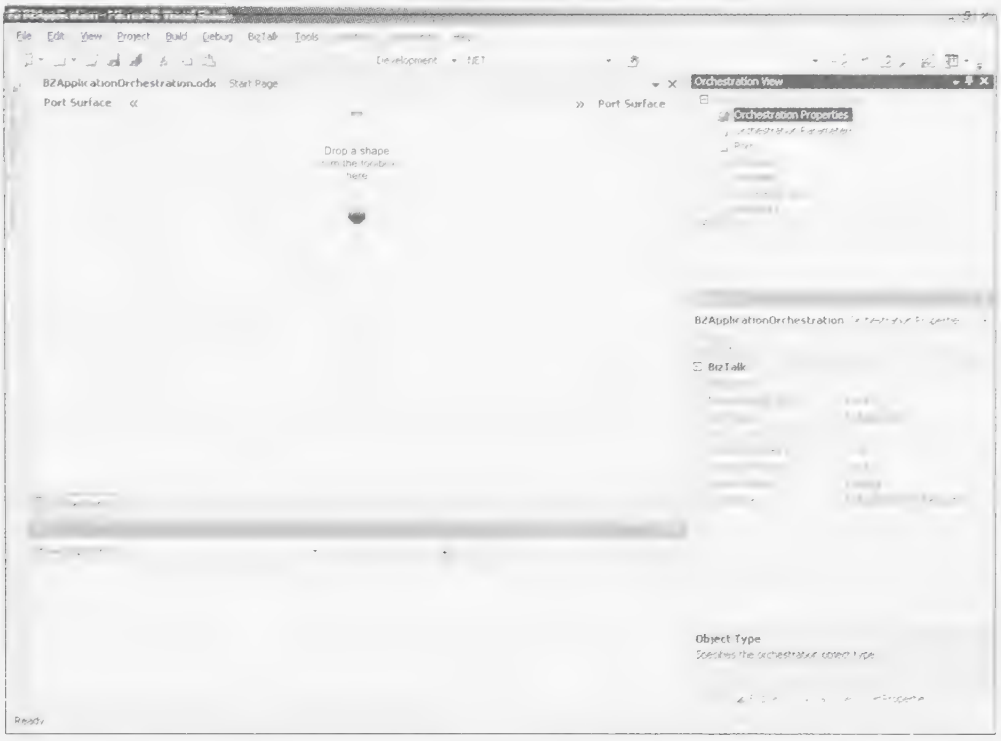


Fig.Biz-3.5

You are now familiar with the method of adding an orchestration to the BizTalk project. After adding an orchestration, we need to add a **message** to the **BZApplication** project. Messages act as a piece of information that is exchanged between organizations. We will discuss the process of adding a message to our **BZApplication** project in the following section.

Message Adding

In BizTalk, messages are simply **XML** or **flat** files used for exchanging information among organizations. BizTalk exchanges XML data through the **Send/Receive** ports and **Send/Receive** locations. The **Send/Receive** locations are used by BizTalk for retrieving messages from one end and forwarding them to the other. BizTalk uses the **Send/Receive** ports for transferring the message from the receive location to the send (destination) location. For more information on the Send/Receive ports, please refer to the 'Introducing Orchestration Designer Shapes' section of Chapter 2.

Messages can be added in **BZApplicationOrchestration** by following these steps:

1. *Right-click* the **Messages** folder on the **Orchestration View** pane to open the context menu.
2. *Select* the **New Message** option from the context menu to add a message to the **BZApplication** project. The **Message_1** message is added to the **Messages** folder, shown encircled in Fig.Biz-3.6.

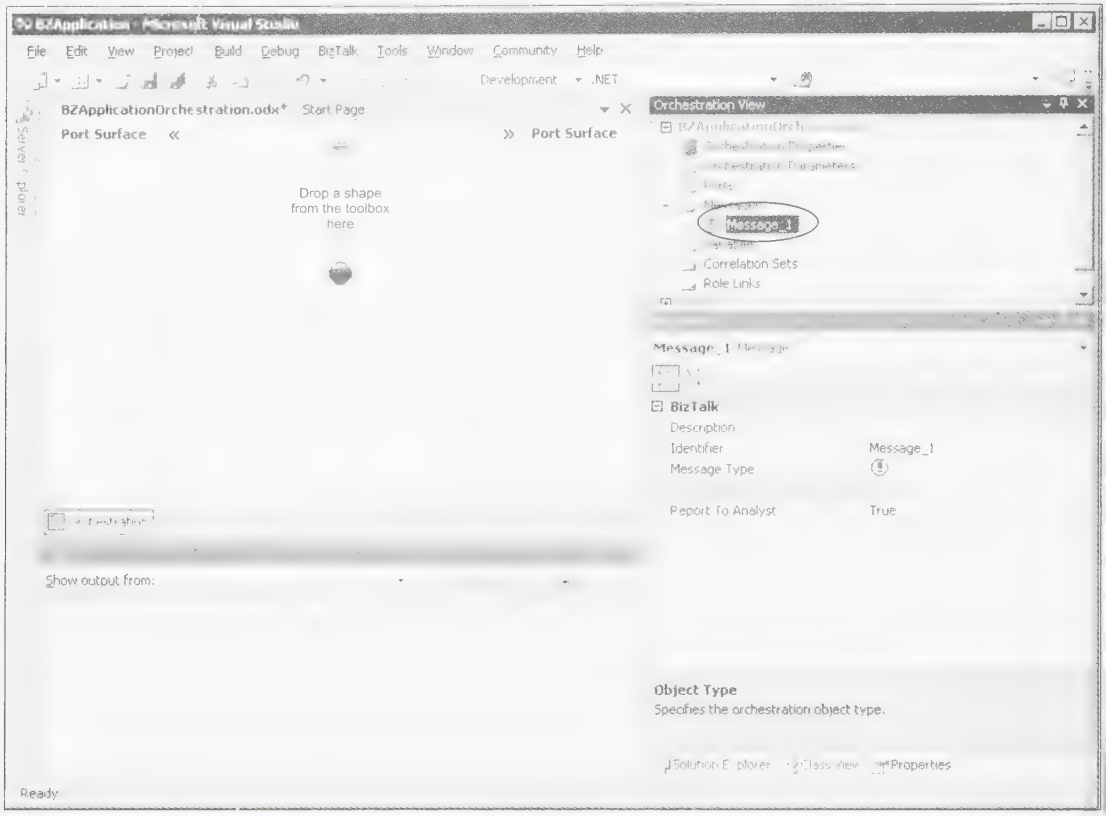


Fig.Biz-3.6

3. *Change* the properties of the **Message_1** message from the **Properties** pane (Fig.Biz-3 according to values mentioned in Table 3.1.

Table 3.1: Properties and values of Message_1 message

Property	Value
Identifier	BZApplicationMessage
Message Type	System.Xml.XmlDocument (in the .Net Classes node)

After applying these message properties, a screen will appear, as shown in Fig.Biz-3 **BZApplicatonMessage** is thus added to your **BZApplication** project.

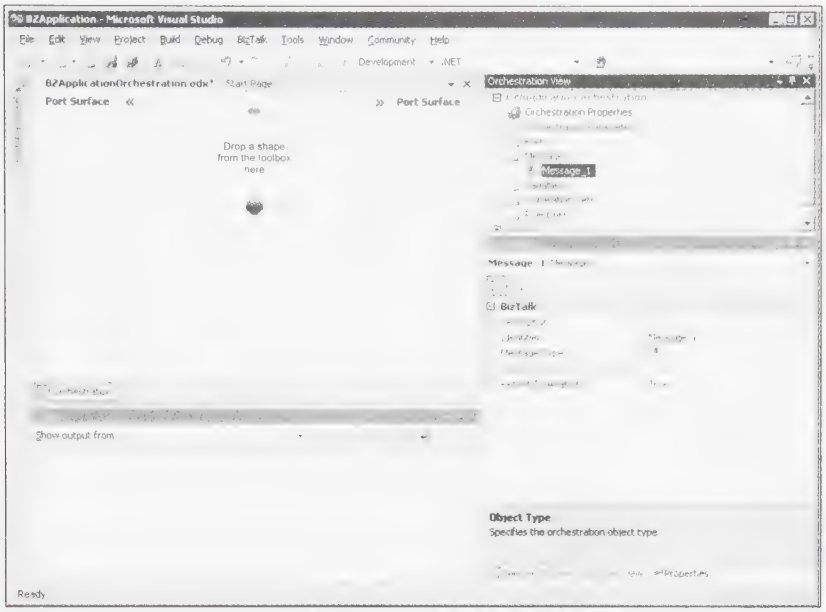


Fig.Biz-3.7

Let's now learn how to add ports in the **BZApplication** project for sending and receiving messages.

Adding Ports in the Project

Ports are communication **endpoints**, which are used by BizTalk Server for sending or receiving messages. The port must be configured to a specific receive location for receiving the message and to a specific send location, which acts as a message destination location. There are **logical** and **physical** ports. Logical ports are required at the time of developing a BizTalk application in Microsoft Visual Studio 2005.

Physical ports are required at the time of accessing the BizTalk application (developed in Microsoft Visual Studio 2005) in BizTalk Server. You need to define both these ports, as they need to be bound together to exchange messages. For more information on the binding of logical and physical ports, please refer to the 'Performing the Binding of Logical and Physical Ports' section further in this chapter.

While defining a port, you need to define the **Port type**, which is either **one-way** or **request-response**. The one-way port type allows messages to be sent from one-way only whereas the request-response type allows sending of message and receiving their acknowledgement when the message has been send to the sender.

Ports can handle different types of messages, such as XML file or flat file. Ports can be added in the **BZApplicationOrchestration** orchestration by following these steps:

1. In the **Orchestration View** pane, *right-click* the **Ports** folder and *select* the **New Port** option from the context menu to add a new port named **Port_1**. You can see the newly added port (**Port_1**) encircled in Fig.Biz-3.8.

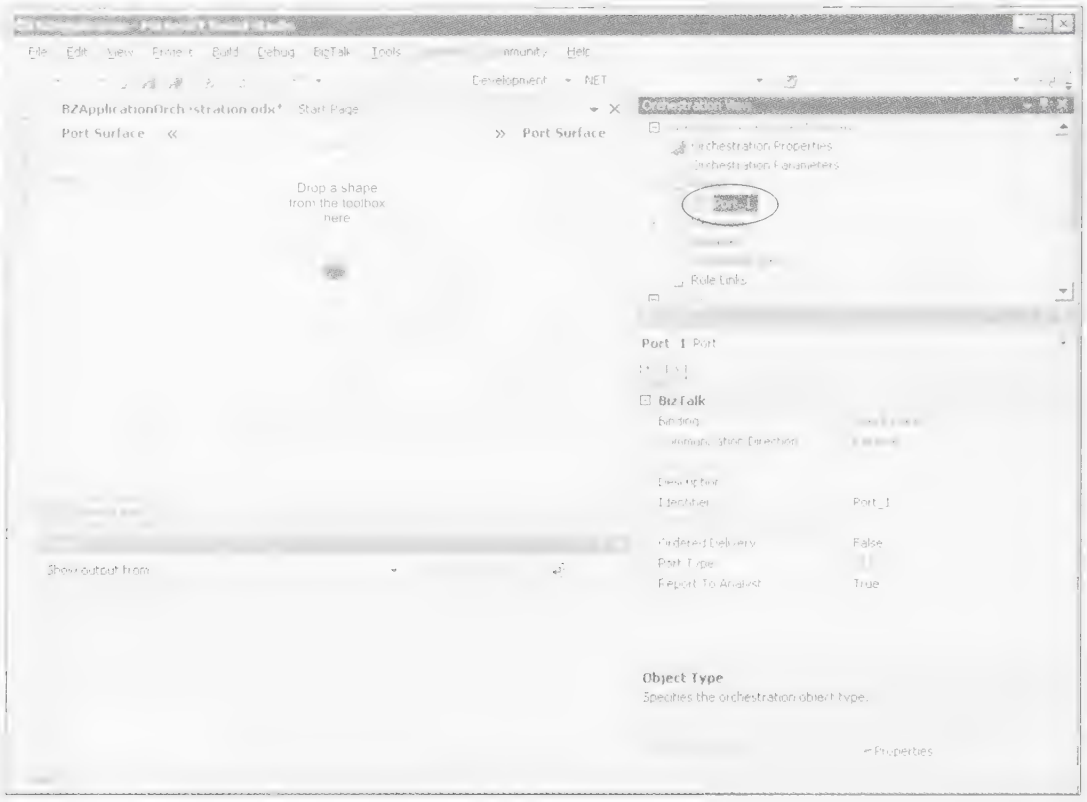


Fig.Biz-3.8

2. Now, *right click* the **Ports** folder in the **Orchestration View** pane to add another port.

In this project, we need two ports. One port will be used to send messages from the Send shape and the other port will be used to receive messages from receive shape. The send/receive shape will be discussed in the 'Adding send/receive shape' section of this chapter.

3. After this, *change* the properties of **Port_1** and **Port_2** ports in the **Properties** pane (Fig.Biz-3.8) according to the values mentioned in the Table 3.2 and Table 3.3.

Table 3.2: Properties for Port_1 port

Property	Value
Identifier	BZApplicationReceivePort
Port Type	Create new One-Way Type (in the Port Types node)
Communication Direction	Receive

When you set the value of Create new One-Way Type as mentioned in Table 3.2, it is necessary to select the **BZApplicationReceivePort** port and set its properties as well (Fig.Biz-3.9).

Table 3.3: Properties for Port_2 port

Property	Value
Identifier	BZApplicationSendPort
Port Type	One-Way Type ((in the Port Types node)
Communication Direction	Send

When you set the value as One-Way Type of port type property as mention in Table 3.3, it is necessary to select the BZApplicationSendPort port and set its properties in the Port Surface area (Fig.Biz-3.10)

Now, the **BZApplicationReceivePort** port and **BZApplicationSendPort** port is added in the Ports node, you will be presented with a screen, as shown in Fig.Biz-3.9. In Fig.Biz-3.9, you can see that **PortType_1** is also added in the **Port Types** node in the Orchestration View pane.

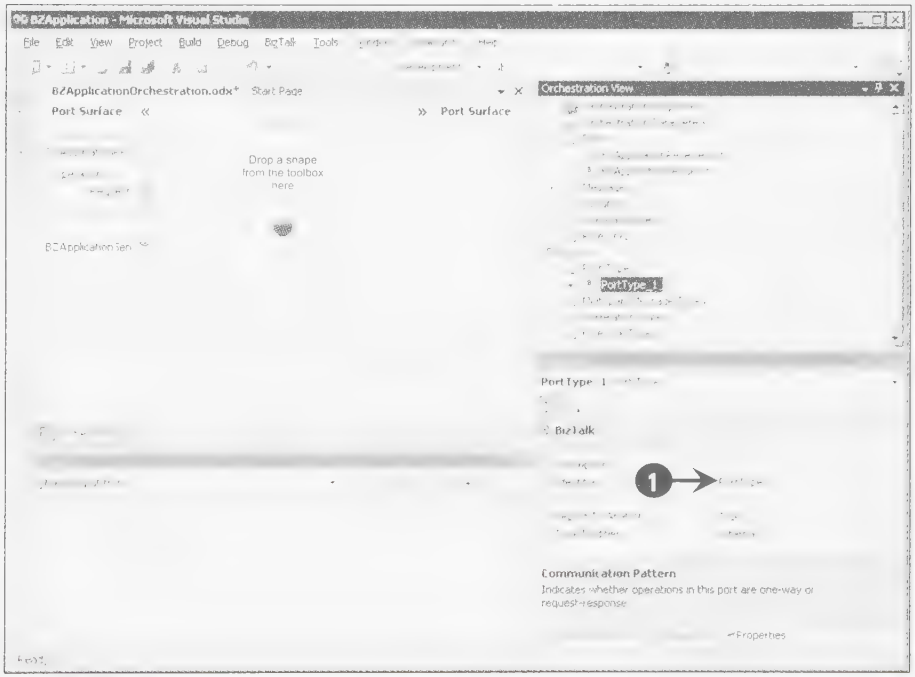


Fig.Biz-3.9

Now, we will define an **Operation** for the port type **PortType_1**.

An operation tells us the tasks that need to be performed by ports. An operation is a part of each port type. An operation has the attributes, namely Request, Response, or both. These attributes are self-explanatory and convey their purpose quite clearly. An operation can be defined by following these steps:

1. First, *change* the **Identifier** property of the **PortType_1** Port type in the Properties pane to **BZApplicationPortType** (Fig.Biz-3.9).

2. Next, *expand* the **BZApplicationPortType** node to open the screen with the added **Operation_1** operation, as shown in Fig.Biz-3.10.

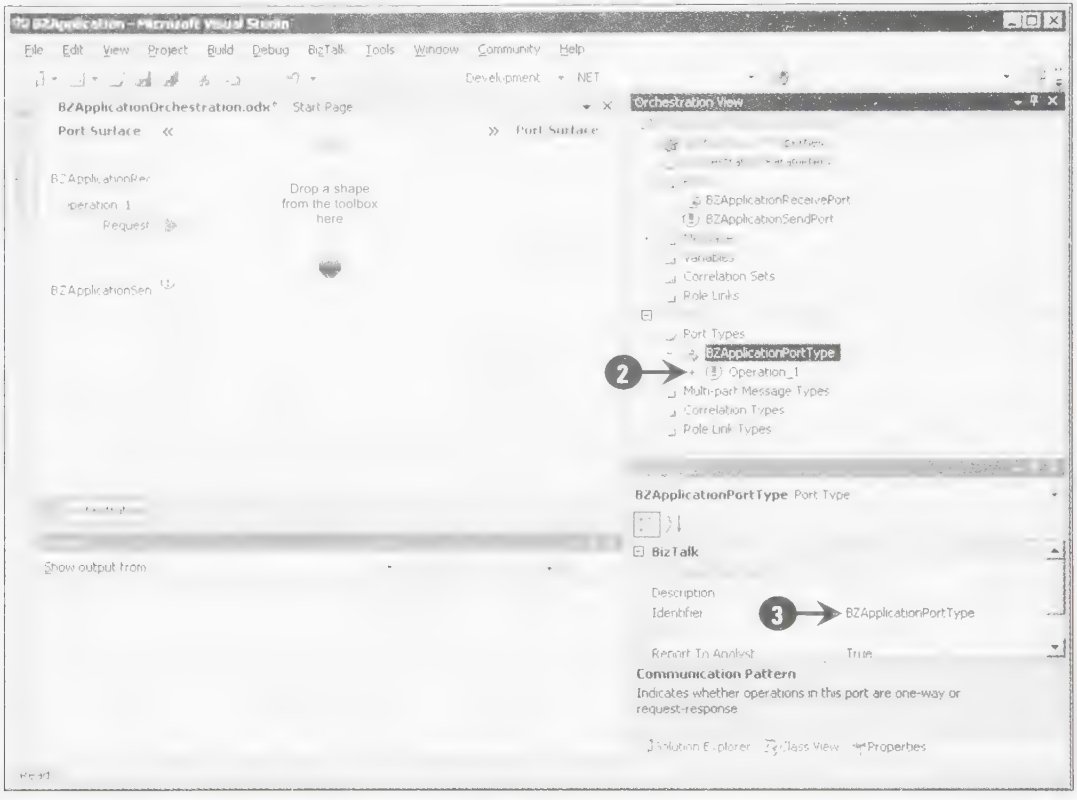


Fig.Biz-3.10

3. *Change* the **Identifier** property of the **Operation_1** operation in the Properties pane to **BZApplicationOperation** (Fig.Biz-3.10).
4. *Expand* the **BZApplicationOperation** operation. This displays the new **Request** attribute (Fig.Biz-3.11).
5. *Select* the **Request** attribute, and *change* **Message Type** property in the **Properties** pane to **System.Xml.XmlDocument** (in the **.NET Classes** node).

As we have already mentioned that the port type is One-Way for BZApplicationSendPort, we don't need to define another port type for BZApplicationSendPort; we will use the BZApplicationPortType port type by selecting the BZApplication.BZApplicationPortType value for port type property of BZApplicationSendPort.

When you complete all these steps correctly, your screen will look like the one shown in Fig.Biz-3.11.

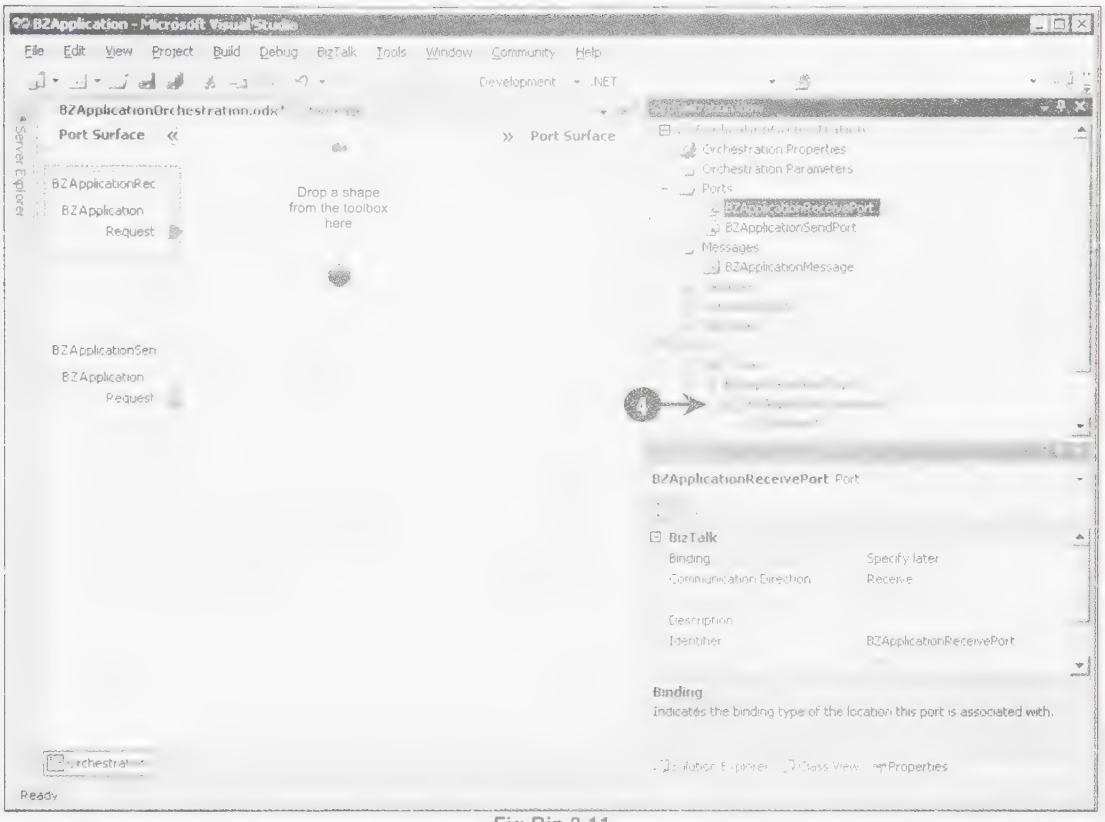


Fig.Biz-3.11

Now, let's add the **Send** and **Receive** shapes to the orchestration for building the business process.

Adding receive/send shapes

The **Receive/Send** shapes enable you to receive/send messages from the **Send/Receive** ports in a business process orchestration. These receive/send shapes are BizTalk objects that are used to communicate with the **Receive/Send** ports for exchanging the message. The **Receive** and **Send** shapes can be added to **BZApplicationOrchestration** by following these steps:

1. *Drag* the **Receive** and **Send** shapes from the **BizTalk Orchestrations** components on the **Toolbox**) onto the Orchestration Surface area where the point labeled **Drop a shape from the toolbox here** is indicated Fig.Biz-3.12. The area is shown encircled in Fig.Biz-3.12.

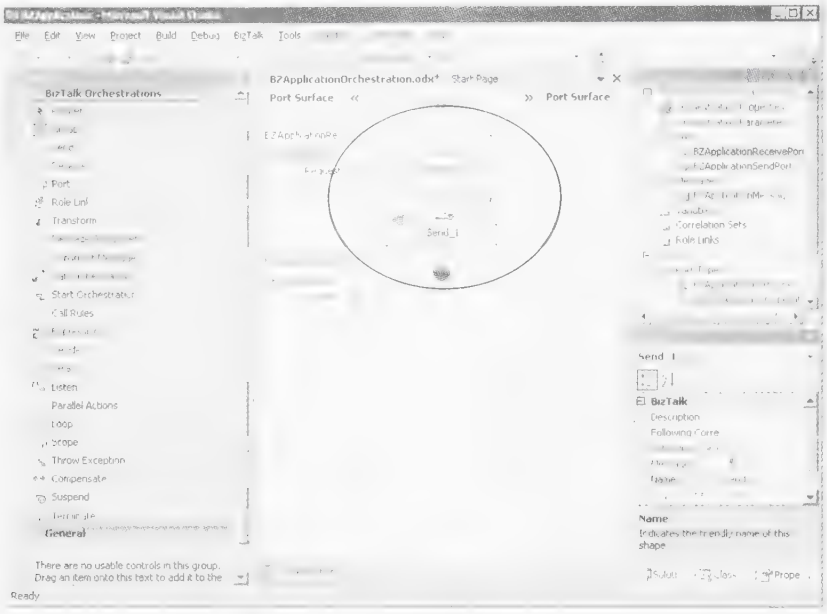


Fig.Biz-3.12

2. *Change* the properties of the **Receive_1** and **Send_1** shapes in the Properties pane according to values in Table 3.4 and Table 3.5.

Table 3.4: Property Values for Receive_1 shape

Property	Value
Activate	True
Message	BZApplicationMessage
Name	BZApplicationReceiveShape
Operation	BZApplicationReceivePort.BZApplicationOperation.Request

Table 3.5: Property Values for Send_1 shape

Property	Value
Message	BZApplicationMessage
Name	BZApplicationSendShape
Operation	BZApplicationSendPort.BZApplicationOperation.Request

After entering these above values, you will be presented with screen, as shown in Fig.Biz-3.13.

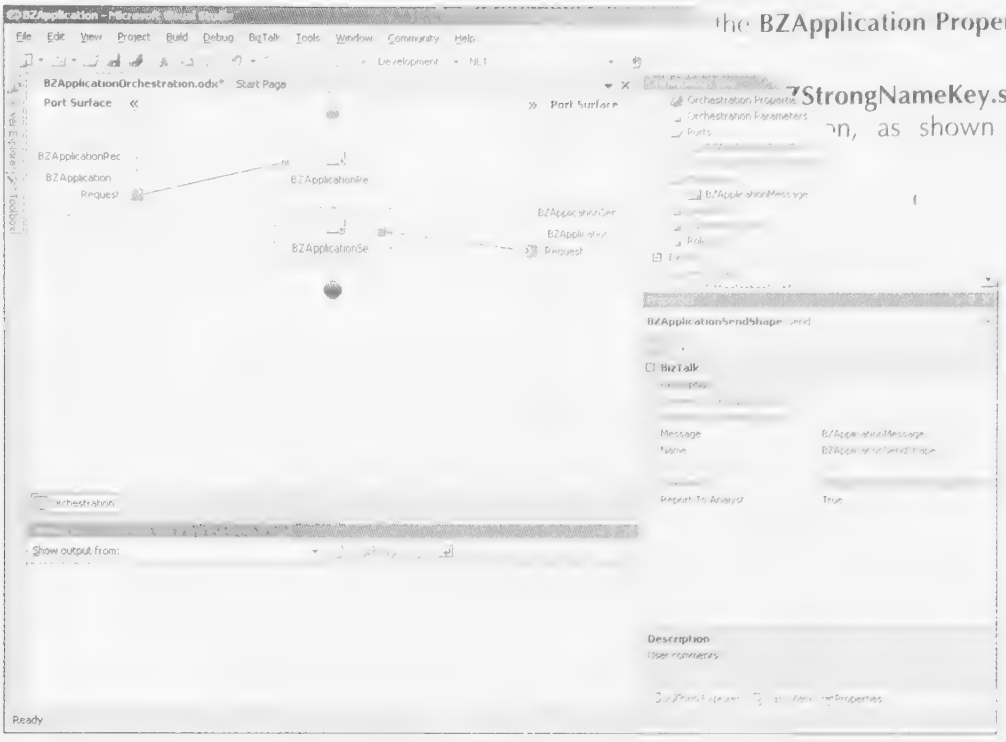


Fig.Biz-3.13

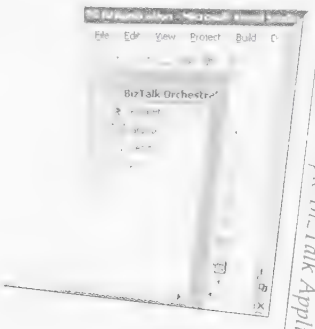
Your application is now ready with ports and different shapes for communication. It is time to deploy the **BZApplication** project. However, before deploying the **BZApplication** project, however, we must first strongly sign the assemblies involved in the project to give them unique identification.

In the next section, we will see how to assign a strong name to the assembly.

Assignment of Strong Naming to the Assembly

In .NET framework, GAC (Global Assembly Cache) is a machine code cache that stores assemblies. These assemblies can be shared by multiple applications on the computer. That is why assemblies need to be strongly signed for their unique identification in the GAC . Assigning a strong name to the assemblies in BizTalk Server will make them easier to identify from GAC at the time of accessing the application from the BizTalk Server.

Strong Name, also referred as "SN", is a technology introduced with the .NET platform. A project's identity name consists of a simple text name, the version number, and the public key and the digital signature. All these are used to assign a strong name to an assembly. During the deployment process of the BizTalk project in Microsoft Visual Studio 2005, each assembly needs to be strongly signed for its unique identification in GAC . You can strongly sign your assemblies by associating the BizTalk project with a strong name assembly key file. BizTalk based projects need to be deployed in GAC. For this, .NET assemblies related to the project need to be strongly signed.



to a strong name, you must first create the 1,024-bit
is part of the owner's digital certificate and is available
otected by and available only to the owner of the key.
nications that use the key are kept secure. You can do
SN.EXE command at the **Visual Studio 2005 command**

ıe **BZApplication** project by following these steps:

and Prompt, *click* **Start→All Programs→Microsoft Visual Tools→Visual Studio 2005 Command Prompt**. The Visual ɔears, as shown in Fig.Biz-3.14.

Fig.Biz-3.14

2. *Enter* the following command at the command prompt to generate the Key Pair, as shown in Fig.Biz-3.15.

```
Sn.exe -k BZStrongNameKey.snk
```

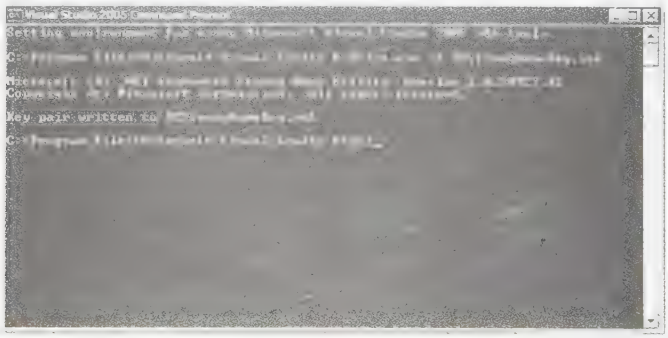


Fig.Biz-3.15

Now the strong name **BZStrongNameKey.snk** is created in your current directory.

Next, you need to assign the strong name **BZStrongNameKey.snk** to your **BZApplication** project. This can be achieved by following these steps:

1. *Open* the **BZApplication** project in Microsoft Visual Studio 2005, if it is closed.
2. *Click* **Project→BZApplication Properties** to open the **BZApplication Property Pages** dialog box, as shown in Fig.Biz-3.16.

3. *Select* the **Assembly** tab under the **Common Properties** node on the **BZApplication Property Pages** dialog box (Fig.Biz-3.16).
4. *Scroll* down in the right pane and *give* the path of the Key file as **C:\BZStrongNameKey.snk** in **Assembly Key File** property value under the **Strong name** section, as shown in Fig.Biz-3.16.

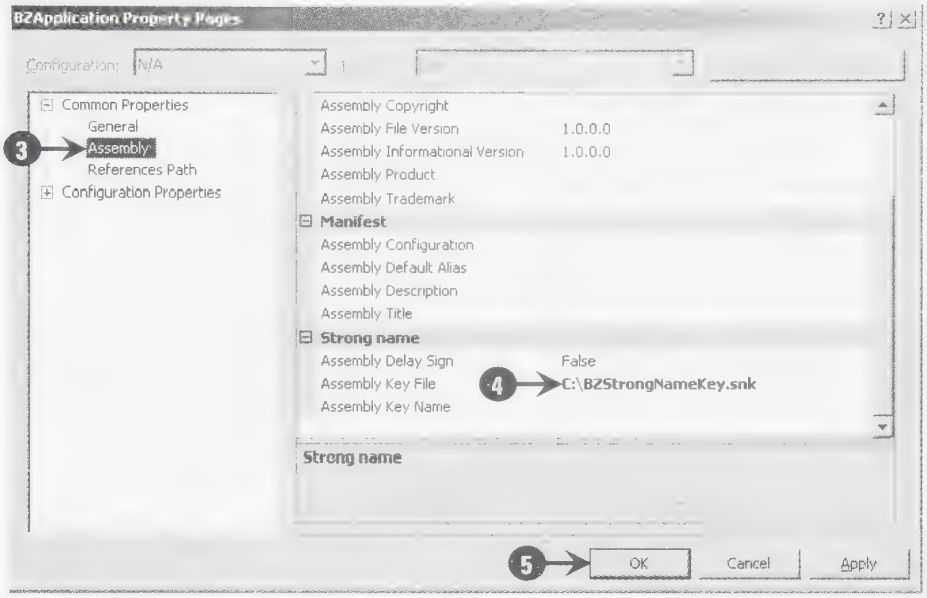


Fig.Biz-3.16

5. *Click* the **OK** button on the **BZApplication Property Pages** dialog box to accept the settings.

We have stored the **.SNK** file in the **C:** drive after generating it. Check your location before assigning the strong key to your BizTalk project. Now, your **BZApplication** project is ready for deployment. We take up this topic in the next section.

Deploying the Project

Once you develop a BizTalk application, the next step is to deploy the project on BizTalk Server 2006. You can deploy your project through **Microsoft Visual Studio 2005**.

The **BZApplication** project can be deployed by following these steps:

1. *Select* **Build→Deploy BZApplication** on the **BZApplication - Microsoft Visual Studio 2005** window.

Once you deploy your project successfully, you will get a successfully deployment with zero (0) error message at the bottom of the output tab of Visual Studio.

When you click the Deploy BZApplication menu option in Microsoft Visual Studio 2005, Microsoft Visual Studio 2005 automatically builds the project first and then deploys it.

Now, it is time to access the **BZApplication** project in BizTalk Server 2006 in order to test it. This testing is required before running the application in the business environment.

Accessing the Application in BizTalk Server 2006

Accessing the BizTalk application involves configuring your BizTalk application (developed in Microsoft Visual Studio 2005) for its working in BizTalk Server 2006. This can be done through **BizTalk Server Administrator Console** or **BizTalk Explorer**. For configuring the application in BizTalk Server, we need to create physical Send/Receive ports and bind them to logical ports (learned earlier). We will now show you how to configure the application with BizTalk Server Administration Console.

Using the Administration Console

BizTalk Server 2006 Administration Console is a tool used for administrating and configuring the BizTalk components including BizTalk based applications. The **BZApplication** project can be configured in BizTalk Server Administration Console by following these steps:

1. *Click* **Start→All Programs→BizTalk Server 2006→BizTalk Server Administration** to open the **BizTalk Server 2006 Administration Console** screen, as shown in Fig.Biz-3.17.

Fig.Biz-3.17

2. *Expand* **BizTalk Server 2006 Administration→BizTalk Group→Applications**, to open the **Applications** node, as shown encircled in Fig.Biz-3.18.
3. *Double-click* the **BZApplication** node in the **Applications** node to *expand* it (Fig.Biz-3.18).

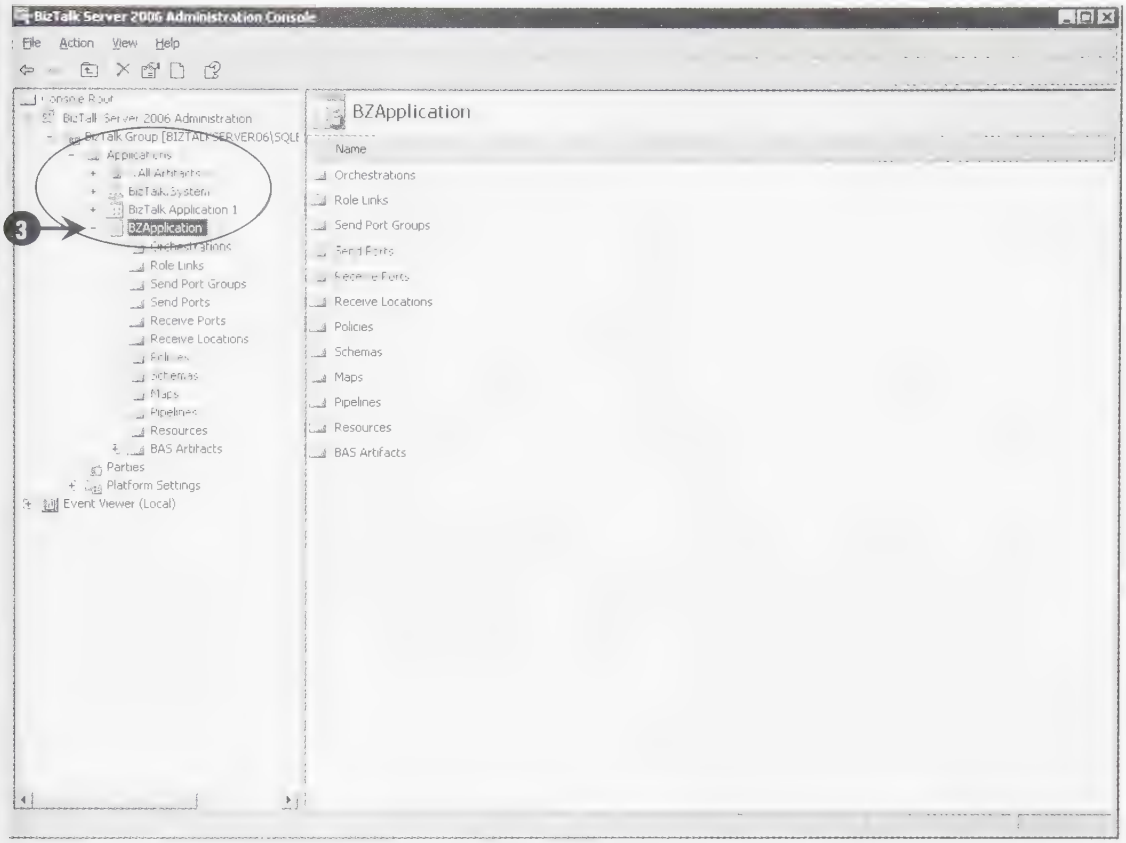


Fig.Biz-3.18

As you now have access to **BZApplication** project, let us move on to create physical Send/Receive ports and locations, which are required for configuring the application in BizTalk Server using the BizTalk Server 2006 Administration Console. These part is also require for exchanging message after binding of physical and logical ports.

Creating a Physical Receive Port and Location

BizTalk Server 2006 Administration Console helps in binding the logical ports with the physical ports. We usually do this at the time of configuring the application with BizTalk Server 2006 Administration Console as this gives us the flexibility to change the source of incoming messages without recompiling the application in Visual Studio. However, before binding the ports, you need to define the send/receive physical ports and their locations. Let us see how to define the receive physical port and its location. This can be done by the following steps:

1. *Right-click* the **Receive Ports** folder of the **BZApplication** node (refer to Fig.Biz-3.18) and
2. *Enter* the value as **BZApplicationReceivePort** in the **Name** textbox (Fig.Biz-3.19).

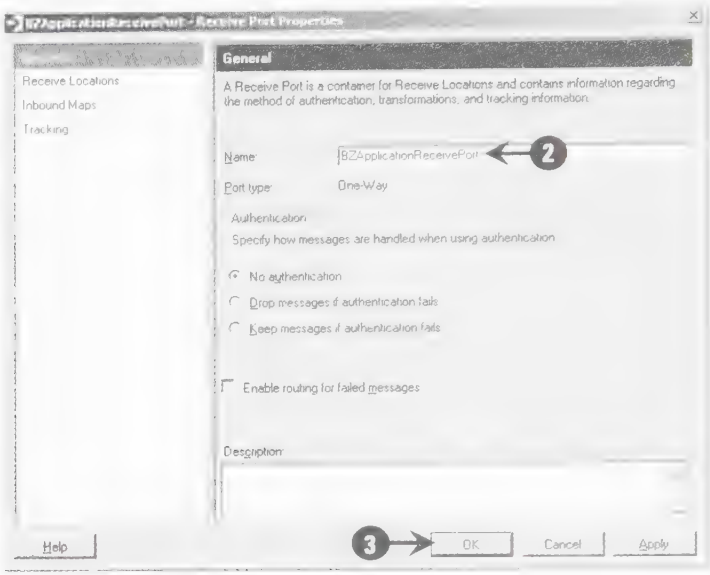


Fig.Biz-3.19

3. *Click the* **OK** button in Fig.Biz-3.19 to accept the settings and to return to **BizTalk Server 2006 Administration Console**.

If the **BZApplicationReceivePort** Receive port is added after clicking the **OK** button, it means that the Receive Physical port is configured. Now, follow these steps to configure the Receive location:

1. *Right-click* the **Receive Location** folder of **BZApplication** node (refer to Fig.Biz-3.18) and *select* **New→One Way Receive Location** from the context menu to open the **Select a Receive Port** dialog box , as shown in Fig.Biz-3.20.

Fig.Biz-3.20

2. *Select* the **BZApplicationReceivePort** from the **Select a Receive Port** dialog box and *click* **OK** button to accept the settings and to *open* the **Receive Location Properties** dialog box.

3. *Enter* the information (mentioned in Table 3.6) in the **Receive Location Properties** dialog box, to have a dialog box as shown in Fig.Biz-3.21.

Table 3.6: Receive location property information

Property	Value
Name	BZApplicationReceiveLocation
Transport Type	File
Receive handler	BizTalkServerApplication
Receive pipeline	PassThruReceive

Pipelines are used for transforming messages at the time of message receiving and sending. In this example, we have used the PassThruReceive pipeline as we do not require message processing; this is required when message destination is known and no validation, encoding and disassembling is required.

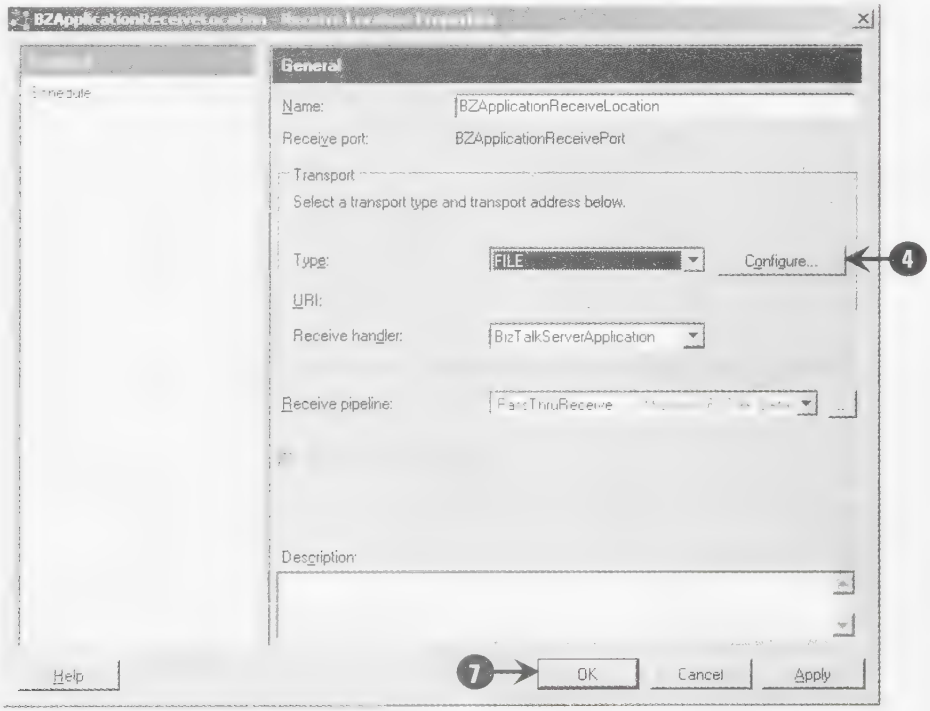


Fig.Biz-3.21

4. *Click* the **Configure** button on the **Receive Location Properties** dialog box to configure the FILE format and to open the **FILE Transport Properties** dialog box, as shown in Fig.Biz-3.22.
5. *Browse* the **Receive folder** location from where the server will receive messages. In our case, the **Receive folder** location is **C:\receive** (Fig.Biz-3.22).

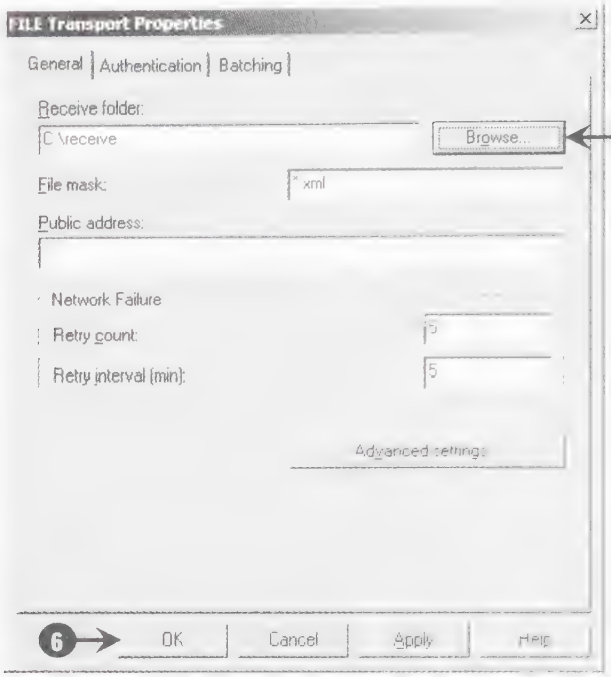


Fig.Biz-3.22

When you first open the FILE Transport Properties dialog box (Fig.Biz-3.22), it shows the Receive folder textbox as blank. In this chapter, we have not shown the default figure as it is not required.

6. *Click* the **OK** button on the **FILE Transport Properties** dialog box to return to the **Receive Location Properties** dialog box. Before *clicking* the **OK** button, ensure that the File Mask textbox value is *.xml.
7. *Click* the **OK** button on the **Receive Location Properties** dialog box (Fig.Biz-3.21) to close the dialog box and return to the **BizTalk Server 2006 Administration Console** window, as shown in Fig.Biz-3.23.

Select the Receive Locations folder from the right pane of BizTalk Server 2006 Administrator Console if the **BZApplicationReceiveLocation** receive location is not displayed on your screen (Fig.Biz-3.23).

By default, the receive location is disabled. To enable this location, right-click **BZApplicationReceiveLocation** on the Receive Locations folder from the right pane of BizTalk Server 2006 Administrator Console and select the **Enable** option from context menu to enable the **BZApplicationReceiveLocation** receive location (Fig.Biz-3.23).

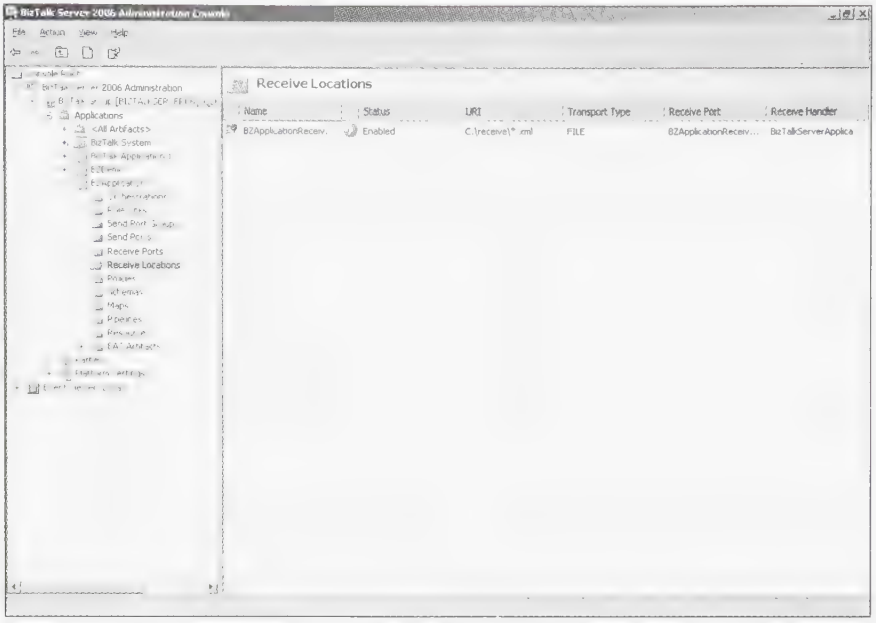


Fig.Biz-3.23

Now, after configuring the Receive location and port, it is time to configure the Send port and location.

Creating a physical send port and location

The Send port is used to send the XML file in the Receive folder (discussed earlier) to the destination mentioned in the configuration of Send port.

A physical Send port can be created by the following steps:

1. *Right-click* the **Send Ports** folder in the **BZApplication** node of BizTalk Server 2006 Administrator Console (refer to Fig.Biz-3.18) to open the context menu.
2. *Select* **New→Static One-Way Sent Port** option from the menu to open the **Send Port Properties** dialog box, as shown in Fig.Biz-3.24.
3. *Enter* the information mentioned in Table 3.7, on the **Send Port Properties** dialog box (Fig.Biz-3.24).

Table 3.7: Send Port Properties Value

Property	Value
Name	BZApplicationSendPort
Transport Type	File
Send handler	BizTalkServerApplication
Send pipeline	PassThruTransmit

The **PassThruTransmit** pipeline is required when no document processing is necessary before the message reaches the destination.

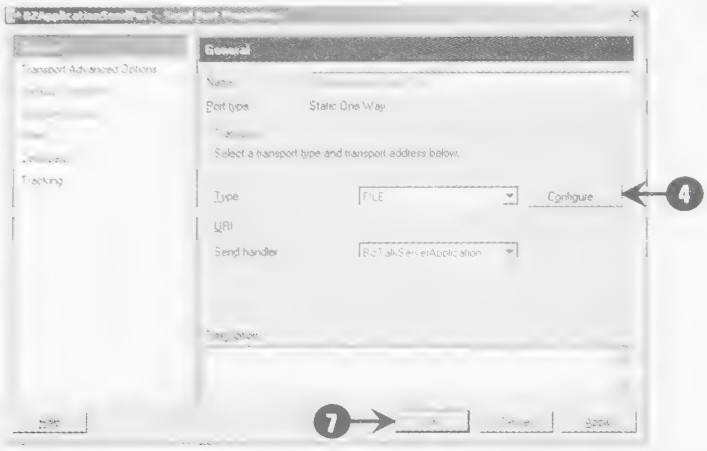


Fig.Biz-3.24

4. Click the **Configure** button on the **Send Port Properties** dialog box, to configure the FILE format and to open the **FILE Transport Properties** dialog box.
5. **Browse** the **Destination folder** location to where the server will send messages. In our case the **Destination folder** location is **C:\sent**, as shown in Fig.Biz-3.25.

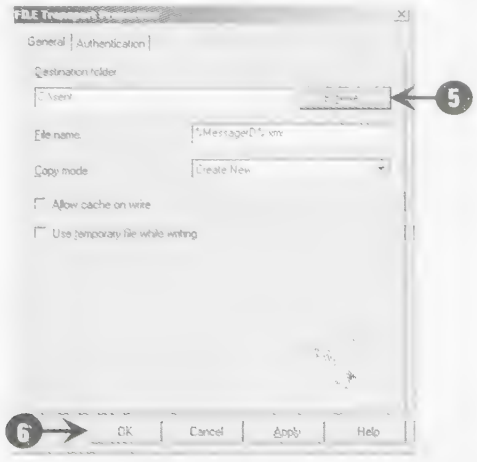


Fig.Biz-3.25

When you first open the **FILE Transport Properties** dialog box, it shows the Destination folder textbox as blank. In this chapter, we have not shown the default figure as it is not required.

6. Click the **OK** button on the **File Transport Properties** dialog box to close the dialog box and return to the **Send Port Properties** dialog box.

Before clicking the OK button, ensure that the File Name textbox value is %MessageID%.xml. This is required to ensure that the messages are stored in xml format with the (%MessageID%.xml naming convention at the destination location mentioned for the Send port.

7. *Click* the **OK** button on the **Send Port Properties** dialog box ((Fig.Biz-3.24) to return to BizTalk Server 2006 Administration Console, as shown in Fig.Biz-3.26

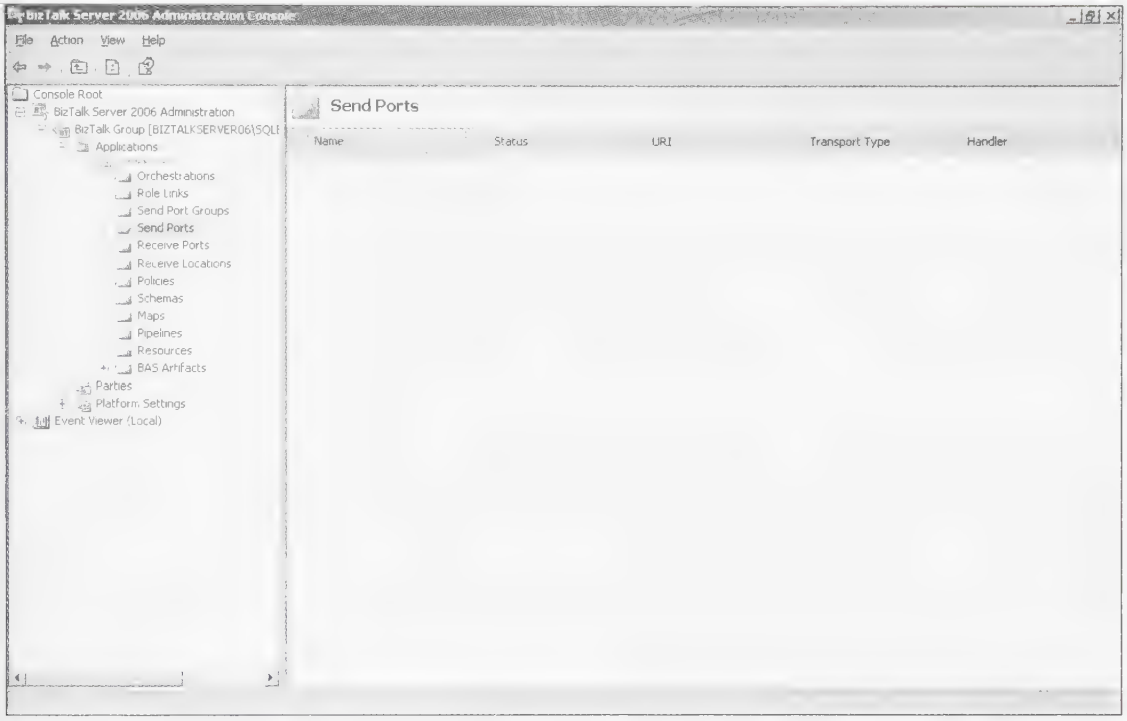


Fig.Biz-3.26

Select the **Send Ports** folder from right side pane of **BizTalk Server 2006 Administrator Console** when the **BZApplicationSendPort** send port will not display as in Fig.Biz-3.26.

By default, Send Ports is not started; its status is **unenlisted** in BizTalk Server 2006 Administration Console. To start this port, right-click the **BZApplicationSendPort** from the right pane of BizTalk Server 2006 Administrator Console and select the **Start** option from the context menu to start **BZApplicationSendPort**

Now, after configuring the receive/send locations and ports, it is time to perform the binding of logical ports with physical ports.

Performing the Binding of logical and Physical port

Logical and physical ports need to be bound for smooth functioning of the business process. This is also required to ensure that the documents are transferred from one location to other without any error.

To bind logical and physical ports, you need to configure the **Orchestration named BZApplicationOrchestration,** developed in **Microsoft Visual Studio 2005** in BizTalk Server 2006 Administration Console. Configuring the application for binding involves the following steps:

1. *Right-click* the **BZApplication** node in the BizTalk Server 2006 Administration Console to open the context menu.
2. *Select* the **Configure** option from the context menu to open the **Configure Application** dialog box, as shown in Fig.Biz-3.27.

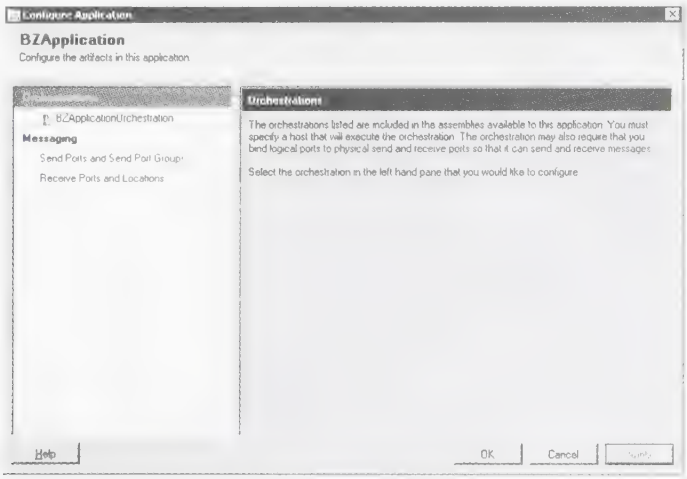


Fig.Biz-3.27

3. *Select* **BZApplicatonOrchestration** from the left pane of the **Configure Application** dialog box to open **BZApplication.BZApplicationorchestration** in the right pane, as shown in Fig.Biz-3.28.
4. *Select* the **Host** value as **BizTalkServerApplication** from the **Host** drop-down list on **BZApplication.BZApplicationorchestration** right pane (Fig.Biz-3.28).

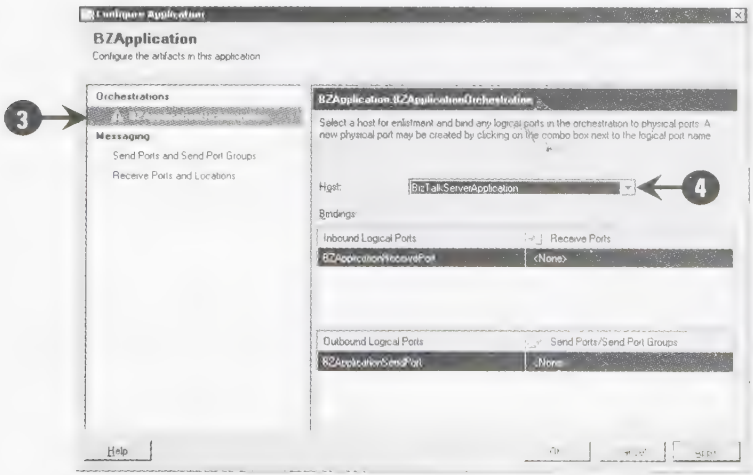


Fig.Biz-3.28

5. *Select* **BZApplicationReceivePort** from the **Receive Ports** drop-down list and **BZApplicationSendPort** from the **Send Port/Send Port Groups** drop-down list in **BZApplication.BZApplicationorchestration** right pane, as shown in Fig.Biz-3.29.

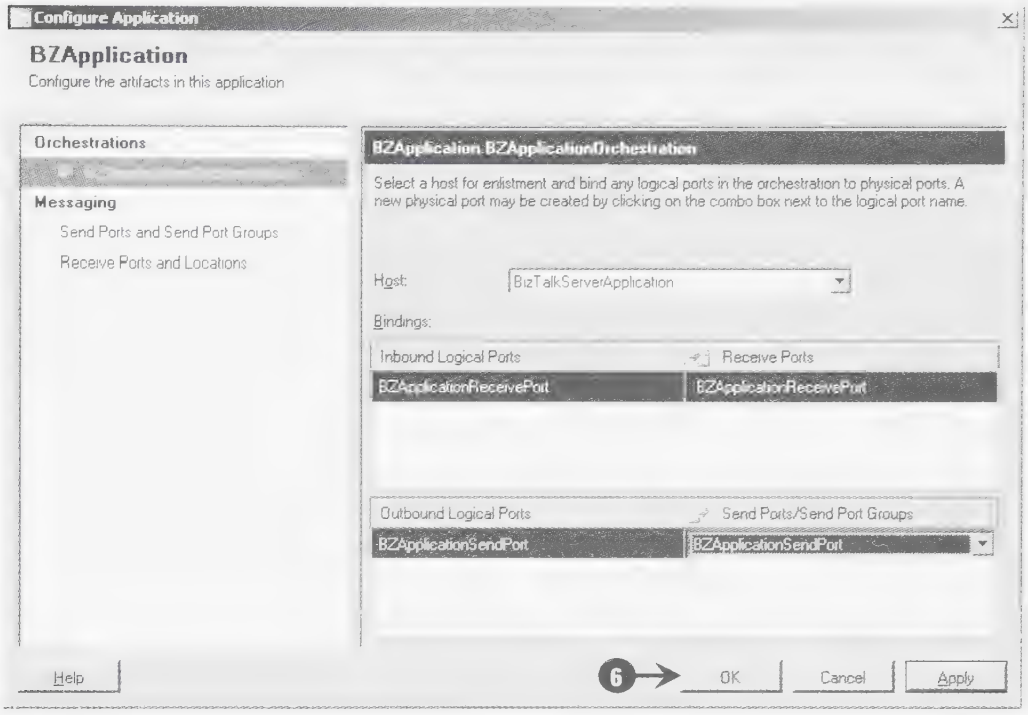


Fig.Biz-3.29

6. *Click* the **OK** button on the **Configure Application** dialog box to close the dialog box and to accept the settings.

You can also add new physical Receive/Send ports and locations from the Configure Application dialog box in case you have not added them before configuring the application for binding. To do this, select the **<New Receive Port...>** option from the Receive Ports drop-down list and the **< New Send Port...>** option from the Send Port/Send Port Groups drop-down list.

Now that your application is configured, it is time to test the application. The process of testing the application is covered in the next section.

Testing the Application

The application can be tested by following these steps:

1. *Right-click* the **BZApplication** node on the BizTalk Server 2006 Administrator Console to open the context menu.
2. *Select* the **Start...** option from the context menu to open the **Start 'BZApplication' Application** dialog box, as shown in Fig.Biz-3.30.

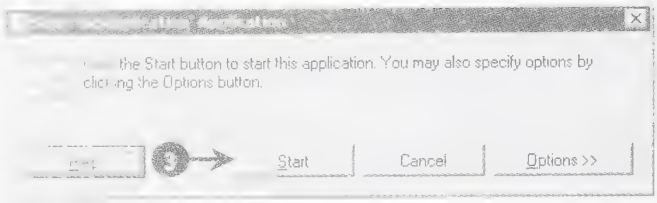


Fig.Biz-3.30

3. *Click* the **Start** button on the **Start 'BZApplication' Application** dialog box to start the application.
4. Now, *copy* an XML file (Test **Message.xml** in our case, you can use your own XML file) to

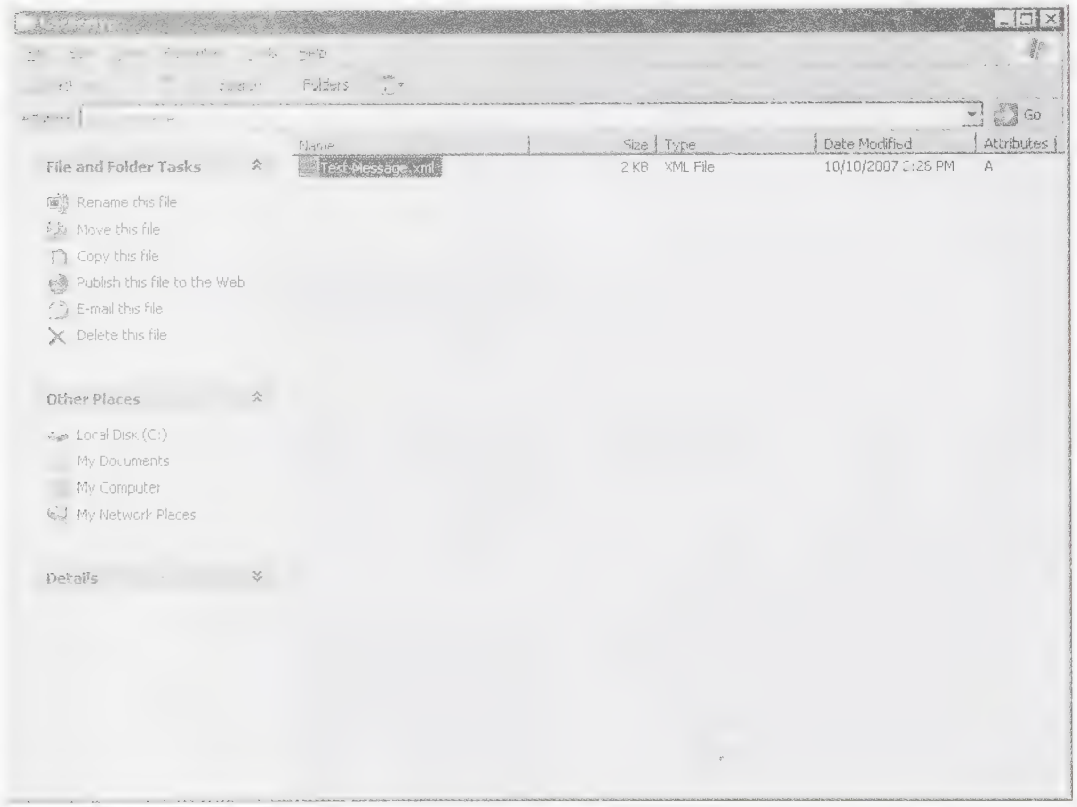


Fig.Biz-3.31

5. After few second, *check* the file in the **Destination** folder, that is **C:\sent**, as shown in Fig.Biz-3.32.

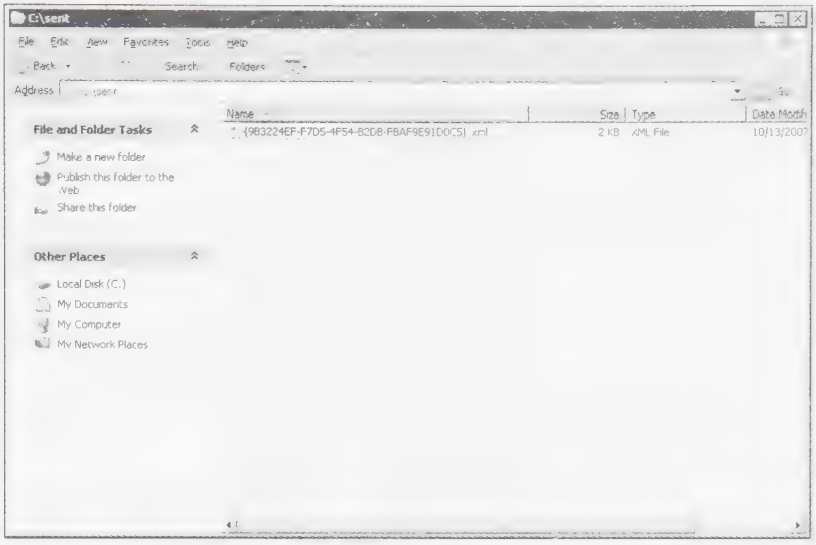


Fig.Biz-3.32

In Fig.Biz-3.32, the file is delivered with the name {9B3224EF-F7D5-4F54-B2DB-FBAF9E91D0C5}.xml. While delivering the file, BizTalk changes the file name with the file name specified in the send port configuration (see Fig.Biz-3.25).

This delivery ensures that your application is configured properly on BizTalk Server 2006 and is easily accessed by the server.

After testing the application, you can stop the application by following these steps:

1. *Right-click* the **BZApplication** node in BizTalk Server 2006 Administration Console and *select* the **Stop** option from the context menu to open the **Stop 'BZApplication' Application** dialog box, as shown in Fig.Biz-3.33.

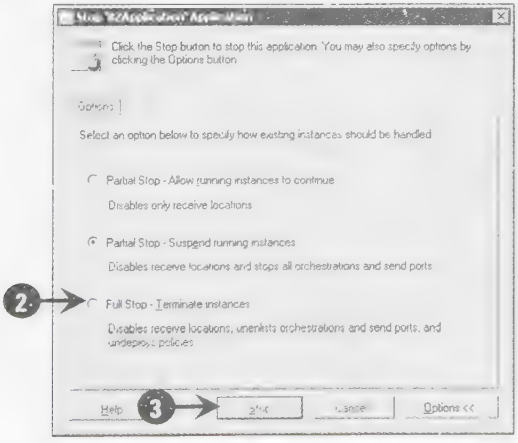


Fig.Biz-3.33

The **Stop 'BZApplication' Application** dialog box has three options, namely:

- **Partial Stop - Allow running instances**: This option allows running instances to continue and disables the receive locations only.
- **Partial Stop - Suspend running instances:** This option suspends running instances and disables receive locations. It stops all orchestrations and send ports.
- **Full Stop - Terminate instances**. This option disables receive locations and unenlists orchestrations and send ports. It also undeploys policies.

In this chapter, we are using the **Full Stop – Terminate instances** option as we are stopping the application.

2. *Select* the radio button beside the **Full Stop – Terminate instances** option (Fig.Biz-3.33).
3. *Click* the **Stop** button to stop the application and to return to BizTalk Server 2006 Administration Console.
4. *Close* the BizTalk Server 2006 Administration Console.

Note

You can also perform the task of configuring, binding and testing the BizTalk application by using the BizTalk Explorer in Microsoft Visual Studio 2005. BizTalk Explorer can be accessed by following these steps:

1. *Open* the BizTalk project in Microsoft Visual Studio 2005.
2. *Deploy* the BizTalk project.
3. Select View→BizTalk Explorer to open BizTalk Explorer, as shown in Fig.Biz-3.34.

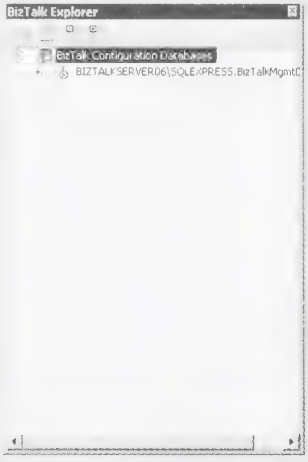


Fig.Biz-3.34

4. Expand the <COMPUTER NAME>\SQLEXPRESS.BizTalkMgmtDB.dbo node in the BizTalk Explorer, to perform the binding of physical and logical ports.

You can perform these steps for opening the BizTalk Explorer. To configure the BizTalk application through BizTalk Explorer, refer to 'Using the BizTalk Explorer' section of Chapter 4.

Now that the application is tested, you can use it for performing your business tasks.

This concludes the chapter. A brief summary of the chapter follows next.

Summary

In this chapter, you have learned about creating a sample BizTalk application with Microsoft Visual Studio 2005. You have also learned how to run the application in BizTalk Server 2006 with BizTalk Server Administration Console by developing the physical ports. In the end, you have learned how to configure the application for binding physical and logical ports. This is done for testing the application in BizTalk Server 2006.

In the next chapter, we are going to develop another project by using BizTalk schemas.

Chapter 4

Implementing Schemas in BizTalk Applications

In this Chapter

- Creating a New Schema Based Project
- Deploying the Project
- Accessing the Application in BizTalk Server 2006
- Testing the Project
- Working with Flat File Schemas
- Adding a Schema to Messages
- Adding Different Shapes to the Project
- Adding a Send Pipeline
- Deploying the Project
- Configuring the Application in BizTalk Server 2006

In today's scenario, organizations need to transfer messages as information to different locations electronically. This information may relate to the vendor-client relationship or it may be some internal data exchange requirement. This kind of information transfer can be done easily and efficiently by using BizTalk Server. You can use BizTalk's predefined schema formats in XML for storing message information. However, you may sometimes need to use the custom XML formats to send and exchange messages. For this, BizTalk provides a functionality of creating your own schema format as per your custom requirement, which you can use for your XML message. BizTalk also has a feature to support flat files. You can use BizTalk to create flat file schemas, read flat files and send them from one location to another.

To understand this concept better, let's take an example of ABC restaurant, which has a number of branches at different locations. Customers come to these branches and place their orders. The branch managers will normally need to send this information along with other information related to the routine functioning of the branch periodically in XML format to their corporate office. However, they face a problem in filtering this information in terms of what is required to be handled at the managerial level and that related to customer orders. To resolve this, ABC restaurant decides to hire a developer to develop a custom BizTalk schema based application that will help save customer information in custom format along with all the information requested by the corporate office. The BizTalk application can then be used to automatically filter the information related to customer orders, thereby simplifying an otherwise laborious and time-consuming process and making it efficient.

Schema is a World Wide Web Consortium (W3C) standard for defining the structure and content of Extensible Markup Language (XML) documents. You can define your own XML documents and the types of data in it.

In this chapter, we are going to develop a BizTalk schema based project. The chapter thus begins by telling you by how to create a new project, as discussed in the previous chapter. This is followed by adding the schema to the project, validating the schema, and generating an instance of the schema file. This is required to check the schema file for errors. Finally, the chapter tells you how to deploy the project with schema based messages and run it on BizTalk Server.

The chapter also shows you the same functionality in the flat file format, which is converted into flat file schema. This helps to convert flat files to XML messages. The user need not have any knowledge of XML to create message files for transmission.

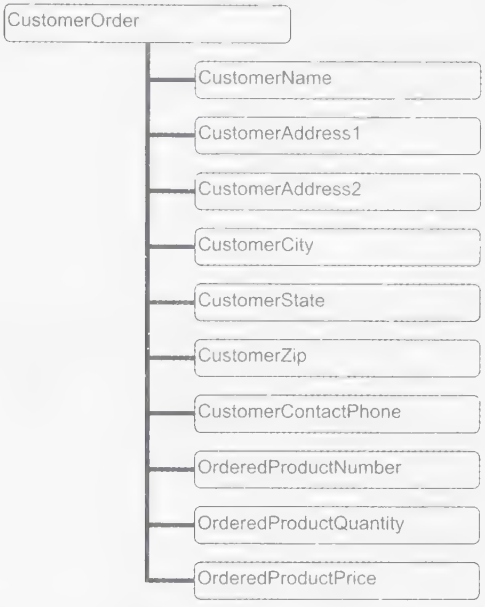
Let us now move forward and take the first step in creating an XML schema based project.

Creating a New Schema Based Project

To build a BizTalk schema based project, we have to first create an XML schema message that will be used to transfer XML documents from one location to another. For this, we have to develop an XML schema to define the structure of the message. You can build the XML schema within Visual Studio 2005 through the BizTalk Schema Editor.

Before developing the BizTalk schema project, let's first discuss the kind of data that you want to thus be associated with a customer's order in a restaurant or hotel. We can define the structure

of this schema (which we will call a customer order schema) used in our project in the following way:



When you have determined the attributes (child fields) of the customer ordering process, such as name of the customer, his address, the price of the products and added them to XML schema, you can say this schema is a customer order schema.

The first step of creating the BizTalk schema based project is to give the project a name. This can be done by following these steps:

1. Start Visual Studio 2005.
2. *Select* **File→New→Project** on Visual Studio to open the **New Project** dialog box.
3. *Select* **BizTalk Projects** from the **Project types** pane and **Empty BizTalk Server Project** from the **Templates** pane.
4. Enter the project name as **CustomerOrder** beside the **Name** text box and *click* the **OK** button to create the BizTalk schema based project.

To know more about creating a BizTalk project, please refer to the section 'Selecting the Project Type' in Chapter 3.

Creating a schema based project involves the following steps:

- Adding Schema to the Project
- Setting the Data Type
- Promoting a Node
- Validating a Schema in the Project
- Adding an Orchestration
- Adding the Schema to Messages

- Adding Different Shapes to the Project

We will discuss all these steps one by one in this chapter, beginning withadding a schema in our project, that is the **CustomerOrder** project.

Adding Schema to the Project

Schema is an XML file that contains the structure and content of XML documents. You have to add a customer order schema according to hierarchy defined earlier in this chapter. Customer order schema can be added to the **CustomerOrder** project by following these steps:

1. *Right-click* the **CustomerOrder** project in Solution Explorer to open the context menu and *Select* **Add→New Item** from the **CustomerOrder** context menu. The **Add New Item - CustomerOrder** dialog box appears, as shown in Fig.Biz-4.1

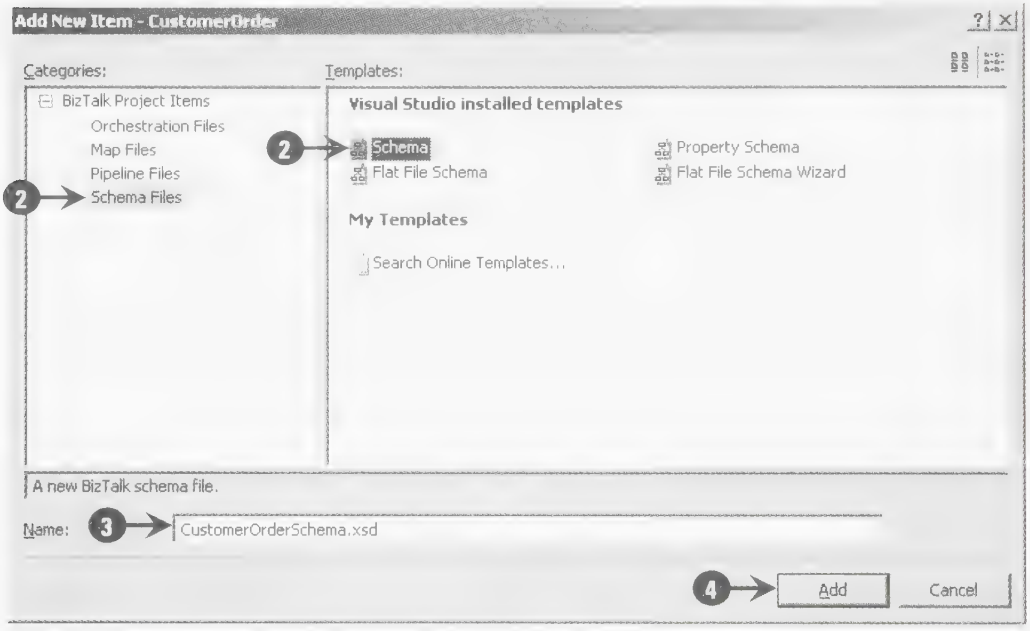


Fig.Biz-4.1

2. *Select* **Schema Files** from the **Categories** pane and **Schema** from the **Templates** pane, as shown in Fig.Biz-4.1.
3. Next, enter the name of the schema file as **CustomerOrderSchema.xsd** beside the **Name** text box (Fig.Biz-4.1).
4. Now, *click* the **Add** button (Fig.Biz-4.1) to add the schema to your project, as shown in Fig.Biz-4.2.

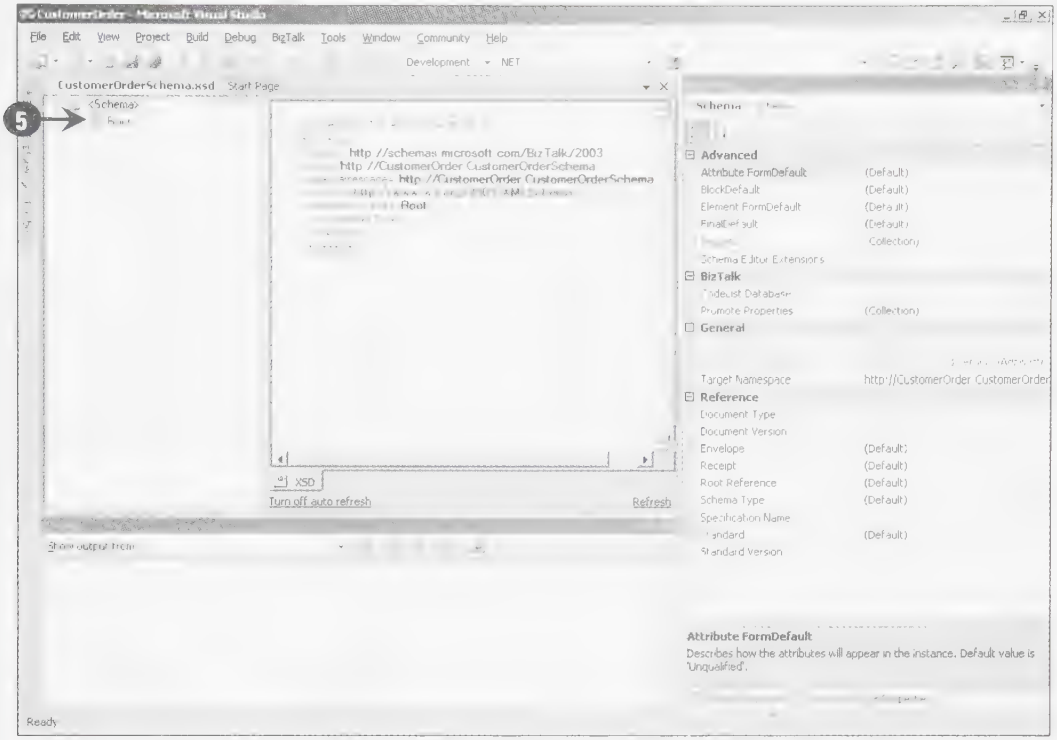


Fig.Biz-4.2

5. *Right-click* the **Root** node of **CustomerOrderSchema.xsd**, and *Select* the **Rename** option from the context menu. Now, rename the name **Root** as **OrderService**.

We will add the various child field elements under the **OrderService** root node.

Thus, the **CustomerOrderSchema** schema is added to the **CustomerOrder** project. Now it is time to insert the child field elements in the **OrderService** root node. A child field element is an XML element that describes items or data in string or number format. Some of the child field elements of our **OrderService** are CustomerName, CustomerAddress1, CustomerAddress2, and so on. The child field elements can be added in the **OrderService** root node of our project by following these steps:

1. *Right-click* the **OrderService** node and *Select* **Insert Schema Node→Child Field Element** from the context menu. A **Field** node appears under the **OrderService** node.
2. *Right-click* the **Field** node and *Select* the **Rename** option from the context menu. Rename this field as **CustomerName**, as shown in Fig.Biz-4.3.
3. Now the **CustomerName** field is added to the **OrderService** root node. Next, repeat steps 1 and 2 to add the following child fields to the **OrderService** root node:
   - CustomerAddress1
   - CustomerAddress2
   - CustomerCity

- CustomerState
- CustomerZip
- CustomerContactPhone
- OrderedProductNumber
- OrderedProductQuantity
- OrderedProductPrice

Once you have added all these child fields to the OrderService root node, your screen wi appear as shown in Fig.Biz-4.3.

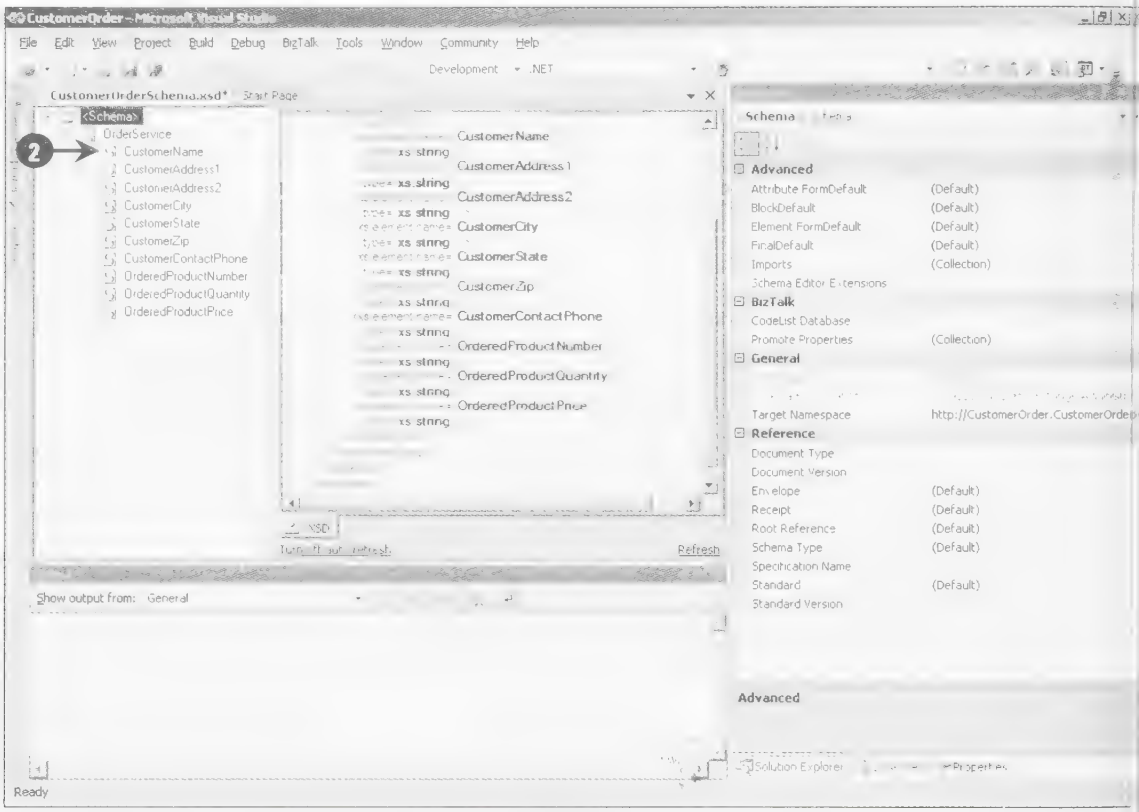


Fig.Biz-4.3

Figure Fig.Biz-4.3 shows the **CustomerOrderSchema** schema with the root node **OrderServic** which contains child fields such as CustomerName, CustomerAddress1, CustomerAddress CustomerCity, CustomerState and CustomerZip. After adding the child fields in the **OrderServi** schema, the next task is to change the data type of the **OrderedProductQuantity** a **OrderedProductPrice** child fields in the **OrderService** schema.

'etting the Data Type

When you have added all the child fields of the **OrderService** root node, the next task is to change the data type of the **OrderedProductQuantity** and **OrderedProductPrice** child fields from string data type (which is assigned by default) to number data type. This is required since we will be performing mathematical operations in them. We can change the data type of these two child fields by the following steps:

1. *Select* **CustomerOrderSchema.xsd** in Solution Explorer to open the **OrderService** schema, as shown in Fig.Biz-4.4.

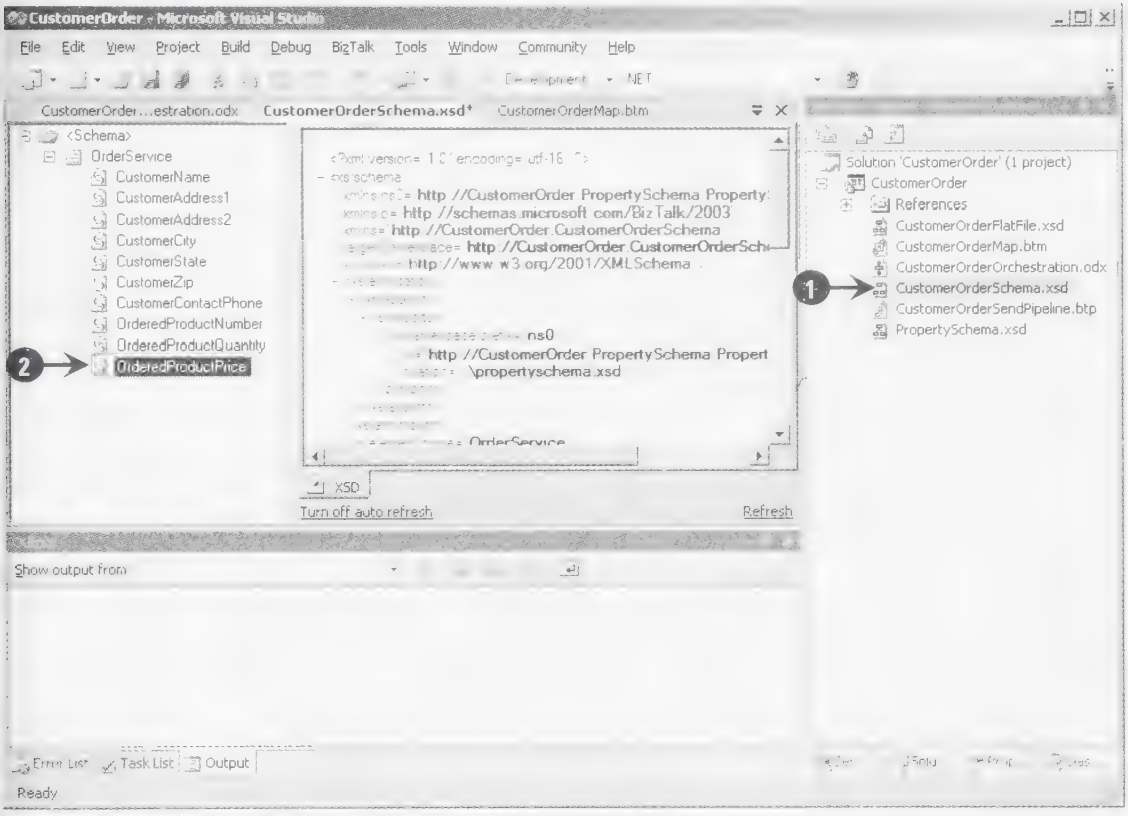


Fig.Biz-4.4

2. *Right-click* the **OrderedProductQuantity** child field from the **OrderService** node and *Select* the Properties option from context menu (Fig.Biz-4.4). The **Properties** window appears, as shown in Fig.Biz-4.5.
3. In the **Properties** window, change **xs:string** to **xs:int** in Data Type drop-down list under the **General** section (Fig.Biz-4.5).

Fig.Biz-4.5

4. Similarly to change the data type of the **OrderedProductPrice** child field, *Right-click* **OrderedProductPrice** and *Select* the **Properties** option from the context menu to open the **Properties** window.
5. In the **Properties** window, change **xs:string** to **xs:decimal** in Data Type drop-down list under the **General** section (Fig.Biz-4.5). Your completed schema appears as shown in Fig.Biz-4.6.

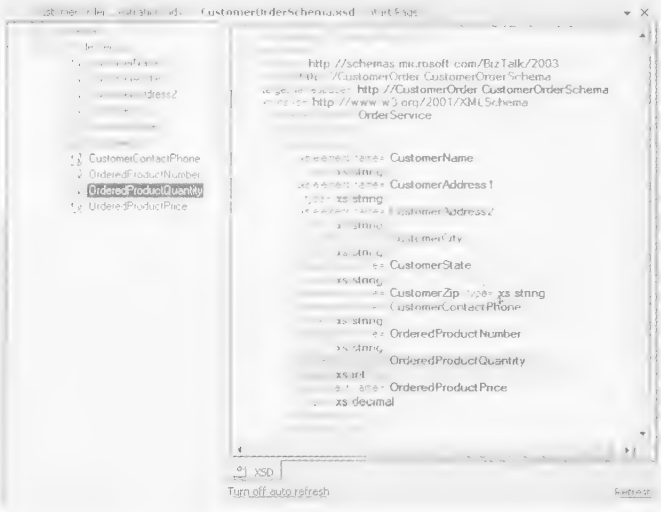


Fig.Biz-4.6

Now that we have assigned the appropriate data types to the child fields of our project, the next task is to promote the nodes (the **OrderedProductQuantity** and **OrderedProductPrice** nodes in our case) that are used to reference a specific value within a message instance. This is taken up in the next section.

Promoting a Node

Promoting a node is a mechanism to reference a specific value within a message instance and make it accessible to the BizTalk Server components, such as orchestrations and messages. Promoting a node in the schema makes the data in the schema available to an orchestration or a message. We have to promote only the **OrderedProductQuantity** and **OrderedProductPrice**, child fields, whose values we will check using the Decide shape and make them accessible with a message. To know how to check these two nodes, please refer to the 'Adding a Decide shape' section of this chapter.

Promoting is also used to check that the values of the **OrderedProductQuantity** and **OrderedProductPrice** nodes are not less than 0 (zero) when the order placed by a customer is entered. To ensure this, follow these steps:

1. *Open* **CustomerOrderSchema.xsd** in Solution Explorer by *selecting* **CustomerOrderSchema.xsd.** (Fig.Biz-4.4)
2. *Right-click* **OrderedProductQuantity** and *Select* **Promote→Quick Promotion** from the context menu to promote **OrderedProductQuantity** (Fig.Biz-4.6.) A **Microsoft Visual Studio** dialog box appears, as shown in Fig.Biz-4.7.

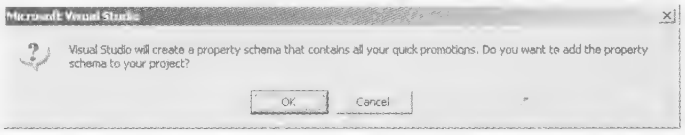


Fig.Biz-4.7

3. *Click* the **OK** button to accept the changes in the **OrderedProductQuantity** node (Fig.Biz-4.7).
4. Similarly, follow the same steps to promote the **OrderProductPrice** node and you will get a screen as shown in Fig.Biz-4.8.

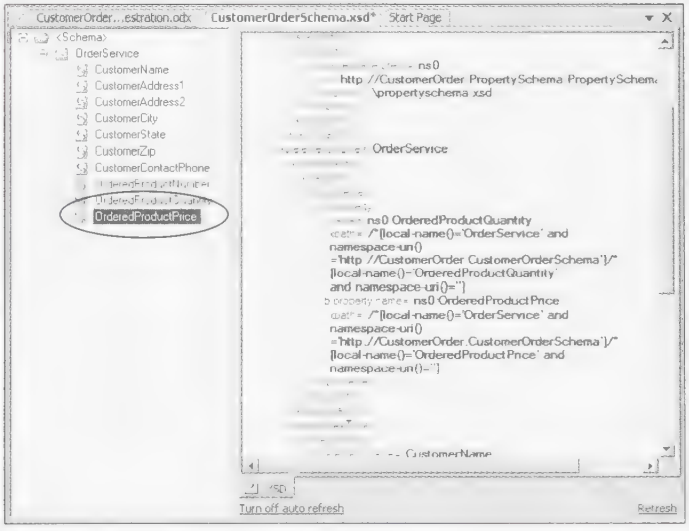


Fig.Biz-4.8

You have now formatted properties for OrderedProductQuantity and OrderProductPrice nodes in simple steps in the previous section. The next step in our project is to validate the schema. This is the subject of our next topic.

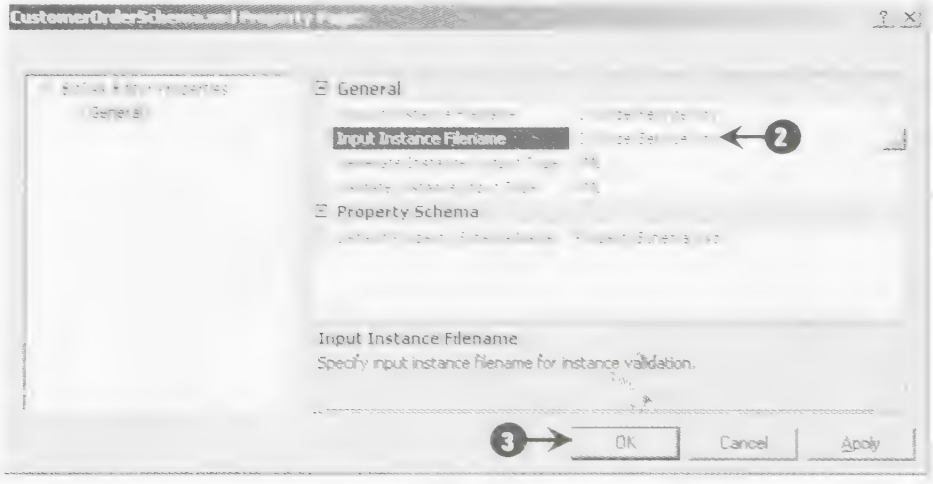
Validating a Schema in the Project

Schema files used in business processes need to be validated to ensure that they maintain consistency in their XML structure. The is done to validate the instance message so that the specified schema is in correct format, which means that each child field in schema must open with a tag and close with a tag. For example, in the following code of schema, the value of CustomerName "CustomerName_0" is opened and closed with the <CustomerName> tag.

```
<ns0:OrderService xmlns:ns0="http://Customer.CustomerSchema">
<CustomerName>CustomerName_0</CustomerName>
<CustomerAddress1>CustomerAddress1_0</CustomerAddress1>
<CustomerAddress2>CustomerAddress2_0</CustomerAddress2>
</ns0:OrderService>
```

The CustomerOrderSchema.xsd schema file must be validated before using it in our project. You can validate the schema by following these steps:

1. Right-click the CustomerOrderSchema.xsd node and select the Properties option from the context menu in Solution Explorer. The CustomerOrderSchema.xsd Property Pages dialog box appears, as shown in figure Fiz.Biz-4.9.



Fiz.Biz-4.9

2. To display the validated output XML file, enter the XML filename C:\OrderService.xml beside Output Instance Filename and Input Instance Filename respectively, as shown in Fig.Biz-4.9.
3. Now, click the OK button to close the CustomerOrderSchema.xsd Property Pages dialog box.
4. Next, right-click the CustomerOrderSchema.xsd node in Solution Explorer and select the Validate Schema option from context menu to validate the schema.

You will see a validated instance success message in the **Output** pane, as shown in Fig.Biz-4.10

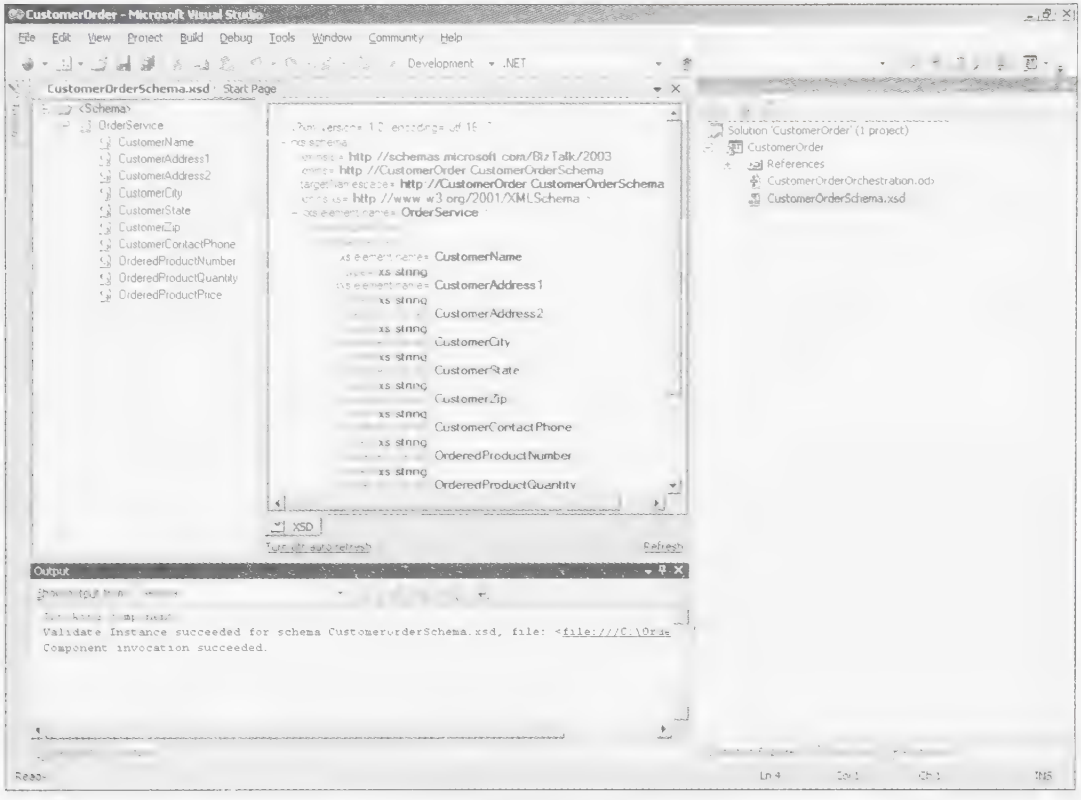


Fig.Biz-4.10

Now that we have validated the **CustomerOrderSchema.xsd** schema file in the **CustomerOrder** project, the next step is to add an orchestration in the project to build the business process with the help of various orchestration designer shapes.

Adding an Orchestration

An orchestration can be added in the CustomerOrder project by following these steps:

1. *Right-click* the **CustomerOrder** project in **Solution Explorer** and *Select* the **Add→NewItem** from the context menu to open the **Add New Item – CustomerOrder** dialog box.
2. *Select* the **Orchestration Files** node in the **Categories** pane and *Select* the **BizTalk Orchestration** node in the **Templates** pane in the **Add New Item – CustomerOrder** dialog box. Also, *enter* the name of the orchestration as **CustomerOrderOrchestration.odx** beside in the **Name** text box.
3. Now *click* the **Add** button to add the **CustomerOrderOrchestration** orchestration item to the **CustomerOrder** project.

For more information on how to add orchestrations, please refer to the 'Adding an Orchestration' section in Chapter 3.

Now that the **CustomerOrderOrchestration** orchestration and the BizTalk **OrderService** schema have been added to the **CustomerOrder** project, it is time to use the **OrderService** schema in the message because the BizTalk project communicates through messages and these messages have to be in XML or flat file format.

Adding the Schema to Messages

As discussed earlier, that BizTalk project communicates with messages and these messages should be in XML format. In this project, we have already developed the OrderService schema (XML document), which we can now use in the message. We now need to add the OrderService schema to the message in the CustomerOrder project. This can be done by the following steps:

1. *Right-click* the **Messages** folder on **Orchestration View**. In addition, *Select* the **New Message** option from the context menu to add a message to the **CustomerOrder** project. Doing this adds the **Message_1** message to the Message folder.
2. *Change* the properties of **Message_1** according to values mentioned in Table 4.1.

Table 4.1: Properties and values of Message_1 message

Property	Value
Identifier	**CustomerOrderMessage**
Message Type	**CustomerOrder.CustomerOrderSchema (in the Schema node)**

After applying these properties, the **CustomerOrderMessage** message is added in the **Messages** folder, as shown in Fig.Biz-4.11.

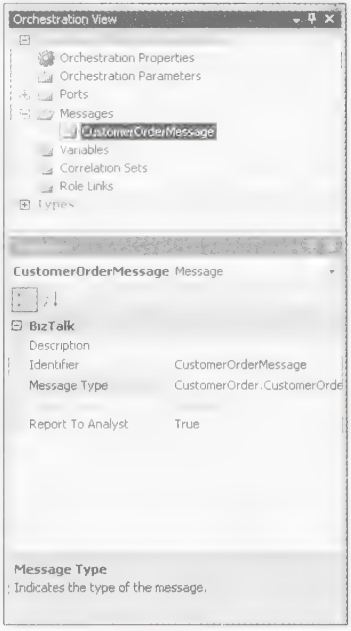


Fig.Biz-4.11

The **CustomerOrderMessage** message is added to the **CustomerOrder** project. Next, let's add the different orchestration shapes to build the business process of the **CustomerOrder** project and to exchange data by using the schema based **CustomerOrderMessage** message.

Adding Different Shapes to the Orchestration

In this section, we will add different orchestration shapes from the Toolbox onto the orchestration surface area to build the **CustomerOrder** business process. We will be adding the following shapes in the **CustomerOrderOrchestration** orchestration:

- Port shapes
- Send and Receive shape
- Decide shape
- Terminate shape

Let's now learn how to add these shapes in the following sections.

Adding Port Shapes in the Project

A Port shape is a logical representation of a message that defines where and how **CustomerOrderMessage** is transmitted in the **CustomerOrder** project. In our project we need to add two Port shapes, the send port (**CustomerOrderReceivePort**) and the receive port (**CustomerOrderReceivePort**). Port shapes can be added in the **CustomerOrderOrchestration** orchestration by following these steps:

1. Drag a Port shape from the Toolbox onto the Port surface area. The **Port Configuration Wizard** dialog box appears, as shown in Fig.Biz-4.12.
2. *Click* the **Next** button on the **Port Configuration Wizard** dialog box (Fig.Biz-4.12) and go to the next stage to set the **Port Properties** of the new port, as shown in Fig.Biz-4.13.

Fig.Biz-4.12

3. *Enter* the port name as **CustomerOrderReceivePort** below the **Name** field (Fig.Biz-4.13).
4. *Click* the **Next** button (Fig.Biz-4.13). This takes you to the next stage of the **Configuration Wizard**, that is ***Select* a Port Type**, as shown in Fig.Biz-4.14. Here, you have to define and set the operations that you want the new port to do.

Fig.Biz-4.13

5. *Select* the **Create a new Port Type** radio button below the ***Select* the port type to be used for this port** option (Fig.Biz-4.14).
6. *Enter* **CustomerOrderPortType** below the **Port Type Name** field (Fig.Biz-4.14).

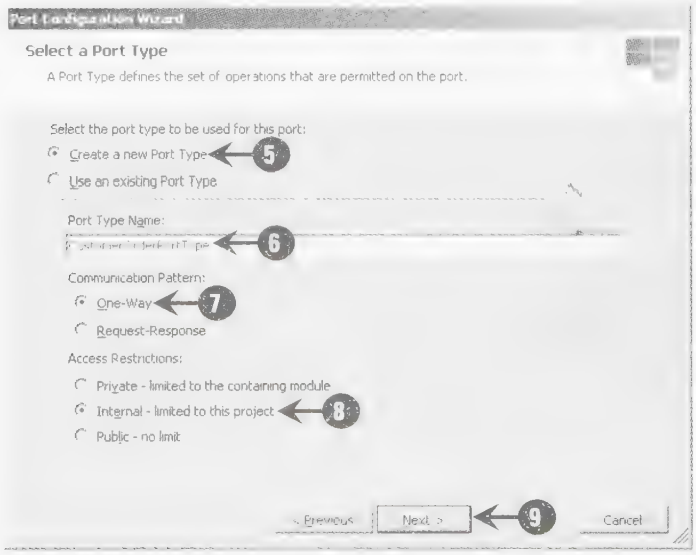


Fig.Biz-4.14

7. *Select* the **One-Way** radio button below **Communication Pattern** (Fig.Biz-4.14).
8. *Select* the **Internal – limited to this project** radio button below **Access Restriction** (Fig.Biz-4.14).
9. Now, *click* the **Next** button (Fig.Biz-4.14). This takes us to the next step of the **Por Configuration Wizard**, as shown in Fig.Biz-4.15. Here, we *Select* the appropriate binding fo the new port.
10. *Select* the **I'll always be receiving messages on this port** option from the **Port direction o communication** drop-down list (Fig.Biz-4.15).

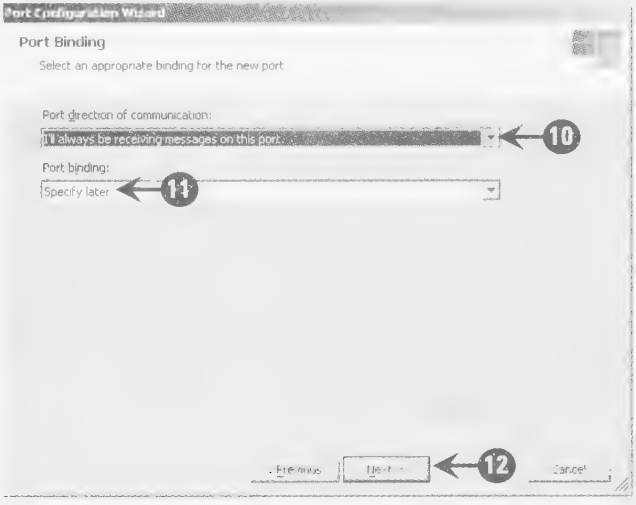


Fig.Biz-4.15

11. *Select* the **Specify later** option from the **Port binding** drop-down list (Fig.Biz-4.15).
12. *Click* the **Next** button to go to the final step of **Port Configuration Wizard** (Fig.Biz-4.15).

Fig.Biz-4.16

13. *Click* the **Finish** button to add **CustomerOrderReceivePort** port.

The **CustomerOrderReceivePort** port shape is thus added to the **CustomerOrder** project. This port shape is used to receive messages from the source location. Now we have to add the send port shape to send messages to the destination location. The send port shape can be added in the **CustomerOrder** project by making the following modifications in the steps used to adding the **CustomerOrderReceivePort** port shape discussed earlier:

1. *Enter* the port name as **CustomerOrderSendPort** and *press* the **Next** key (see step 3 of the 'Adding Port Shapes in the Project' section earlier in the chapter).
2. *Select* the **Use an existing Port Type** radio button below **Select the port type to be used for this port**, as shown in Fig.Biz-4.17.
3. *Select* **CustomerOrder.CustomerOrderPortType** port type below **Available Port Type** (Fig.Biz-4.17).

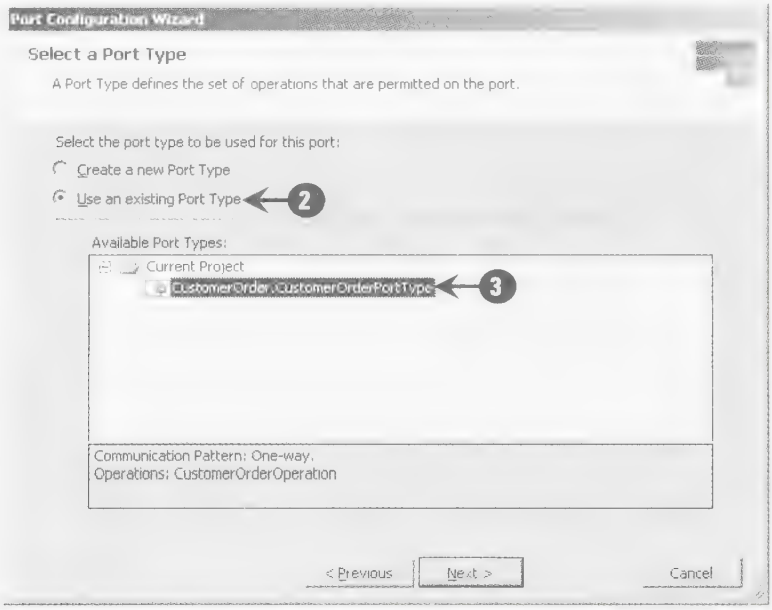


Fig.Biz-4.17

4. *Select* **I'll always be sending messages on this port** option from the **Port direction of communication** drop-down list option (see step 10 and Fig.Biz-4.15 of the 'Adding Port Shapes in the Project' section earlier in the chapter). To complete the process of adding **CustomerOrderSendPort** to our project, follow the remaining steps (11-13) from the said section.

For more information on how to add ports, please refer to the 'Adding Ports in the Project' section in Chapter 3.

Let us now add the Receive and Send shapes in the **CustomerOrder** project. The Receive shape is used to receive messages from a port and the Send shape is used to send a message from a port in the business process orchestration. The Receive and Send shapes are added in **CustomerOrderOrchestration** to build the business process of the **CustomerOrder** project.

Adding the Receive/Send shapes

We need to add the Receive shape and the Send shape in **CustomerOrderOrchestration**. The Receive shape (**CustomerOrderReceiveShape**, in our case) is used to receive message from the **CustomerOrderReceivePort** port and the Send shape (**CustomerOrderSendShape**, in our case) is used to send messages from the **CustomerOrderSendPort** port in the business process orchestration. The Receive and Send shape can be added to the **CustomerOrderOrchestration** orchestration by following these steps:

1. *Drag* the **Receive** and **Send** shapes from the Toolbox in the Orchestration designer surface area where the point labeled Drop a shape from the toolbox here is indicated. The **Receive_1** and **Send_1** shapes are added to your **CustomerOrderOrchestration**.
2. *Change* the properties of the **Receive_1** shape according to the values in Table 4.2.

Table 4.2: Property Values for Receive_1 shape

Property	Value
Activate	True
Message	CustomerOrderMessage
Name	CustomerOrderReceiveShape
Operation	CustomerOrderReceivePort.CustomerOrderOperation.Request

3. *Change* the following properties of the **Send_1** shape according to the values in Table 4.3.

Table 4.3: Property Values for Send_1 shape

Property	Value
Message	CustomerOrderMessage
Name	CustomerOrderSendShape
Operation	CustomerOrderSendPort.CustomerOrderOperation.Request

To know more about how to add the Send and Receive shapes, please refer to the 'Adding receive/send shape' section of Chapter 3.

The **CustomerOrderReceiveShape** and **CustomerOrderSendShape** shapes are thus added to your **CustomerOrderOrchestration**. Next, we will add the Decide shape in the project. The Decide shape is used to add business rules that are discussed in the next section

Adding the Decide Shape

The Decide shape is just like conditional branching that enables you to implement business rules in a business process orchestration. In conditional branching, the flow of the control passes from one operation in a business process to another based on a condition. It is similar to an IF THEN ELSE condition in a programming language. To know more about the Decide shape, please refer to the 'Decide shape' section of Chapter 2.

In our project, we will use the Decide shape to check the **OrderedProductQuantity** and the **OrderedProductPrice** child fields. We already know that when a customer places an order, the values of the **OrderedProductQuantity** and the **OrderedProductPrice** child fields should not and cannot be 0 (zero). In case the value is equal to 0, then the **Else** part of the Decide shape will be executed; otherwise, the main part of the Decide shape (**Rule_1**) is executed (see Fig.Biz-4.18). We write these conditions in the Expression Editor of Decide shape that we will discuss later in this section.

Let us now add the Decide by following these steps:

1. *Drag* the **Decide** shape from the Toolbox onto the Orchestration surface area below **CustomerOrderReceiveShape. Decide_1** is added in the **CustomerOrderOrchestration** orchestration.
2. *Change* **the name** of **Decide_1** to **CustomerOrderDecideShape** from the **Properties** pane, as shown in Fig.Biz-4.18.

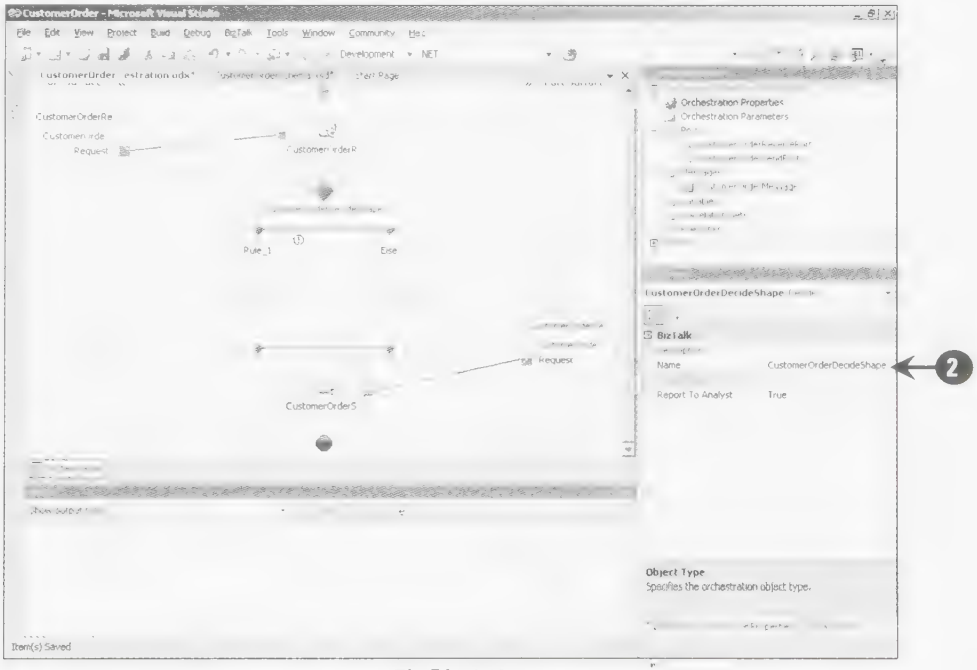


Fig.Biz-4.18

After adding the Decide shape to the **CustomerOrder** project, the next step is to change the properties of the **Rule_1** decision box and implement the business rules in the **CustomerOrderDecideShape**. Business rules are logical expressions used to manipulate the values (perform mathematical, logical computations) contained within the child fields (**OrderedProductQuantity** and **OrderedProductPrice** in our case) of the **CustomerOrderSchema** schema. These rules are used to validate the data of a schema within a specified range or format. We can implement a business rule in **CustomerOrderDecideShape** by following these steps:

1. *Change* the name of **Rule_1** to **CustomerOrder: OrderedProductQuantity not equal to zero** from the **Properties** pane of **Rule_1**, as shown in Fig.Biz-4.19.

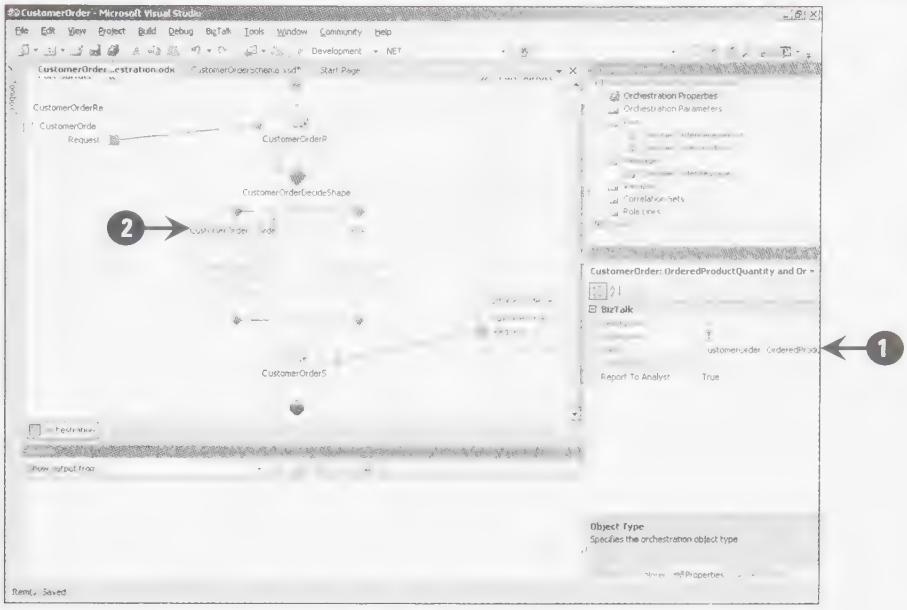


Fig.Biz-4.19

To show the Properties pane of Rule_1, select Rule_1 from the Orchestration designer surface area.

2. *Double-click* **CustomerOrder: OrderedProductQuantity and OrderedProducePrice not equal to zero** from the Orchestration designer area to open the **BizTalk Expression Editor** dialog box, as shown in Fig.Biz-4.120.

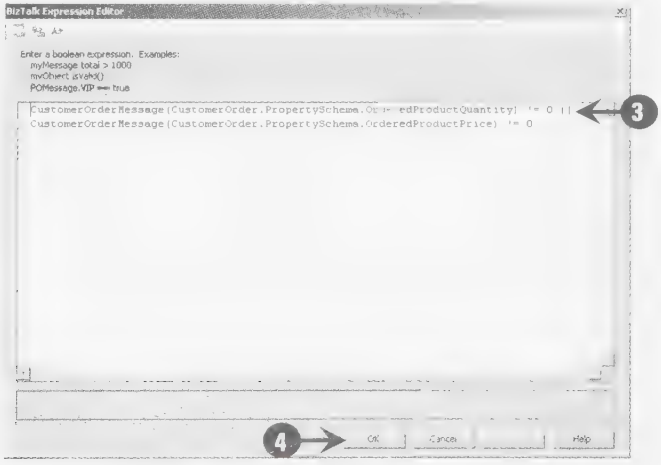


Fig.Biz-4.20

BizTalk Expression Editor enables you to create expressions to expand the capabilities of various orchestration designer shapes. It enables you to assign values to messages or message parts in the Message Assignment shape and manipulate the values of variables in the Decision shape.

3. Now, enter the following Boolean logic in **BizTalk Expression Editor** dialog box:

```
CustomerOrderMessage(CustomerOrder.PropertySchema.OrderedProductQuantity) != 0
|| CustomerOrderMessage(CustomerOrder.PropertySchema.OrderedProductPrice) != 0
```

4. Next, *click* the **OK** button to close the **BizTalk Expression Editor** dialog box.

Now, drag **CustomerOrderSendShape** and drop it below the **CustomerOrder: OrderPropertyQuantity greater than zero** rule of **CustomerOrderDecideShape.** You can see **CustomerOrderSendShape** in the encircle area as shown in Fig.Biz-4.21. This is require, to test the **CustomerOrder** project, when the value of **OrderedProductQuantity** and **OrderedProductPrice** child fields is not equal to zero and all the shape specified below the **CustomerOrderDecideShape** will be executed.

After adding the Decide shape, it is time to add the Terminate shape. The Terminate shape is used to terminate business activities when an error occurs during business process orchestration. Let's learn how to add the Terminate shape in our project.

Adding the Terminate Shape

The Terminate shape is used when an error takes place in a business process. You can use it to immediately end all activities of running an orchestration instance. When an orchestration instance is terminated, an error message is generated that is used to print the error in your application. You can add an error message string to help the administrator to diagnose the situation. The Terminate shape can be added by the following steps:

1. *Drag* the Terminate shape from the Toolbox onto the Orchestration designer surface area below the **Else** decision box. The **Terminate_1** shape is added to your **CustomerOrderOrchestration**.
2. *Change* the name of **Terminate_1** to **CustomerOrderTerminateShape** from the **Properties** pane, as shown in Fig.Biz-4.21.

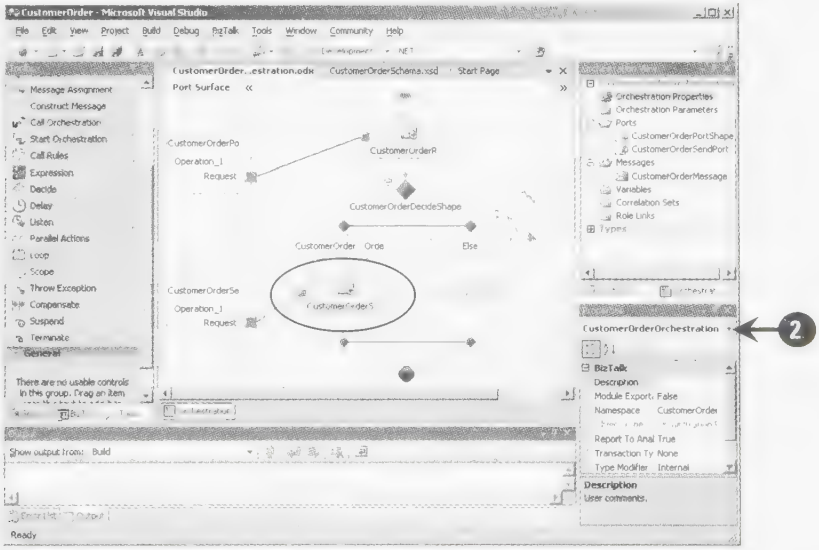


Fig.Biz-4.21

The error message can be added to the **CustomerOrderTerminateShape** by the following steps:

1. *Double-click* **CustomerOrderTerminateShape** from the Orchestration designer area to open the **BizTalk Expression Editor** dialog box, as shown in Fig.Biz-4.22.

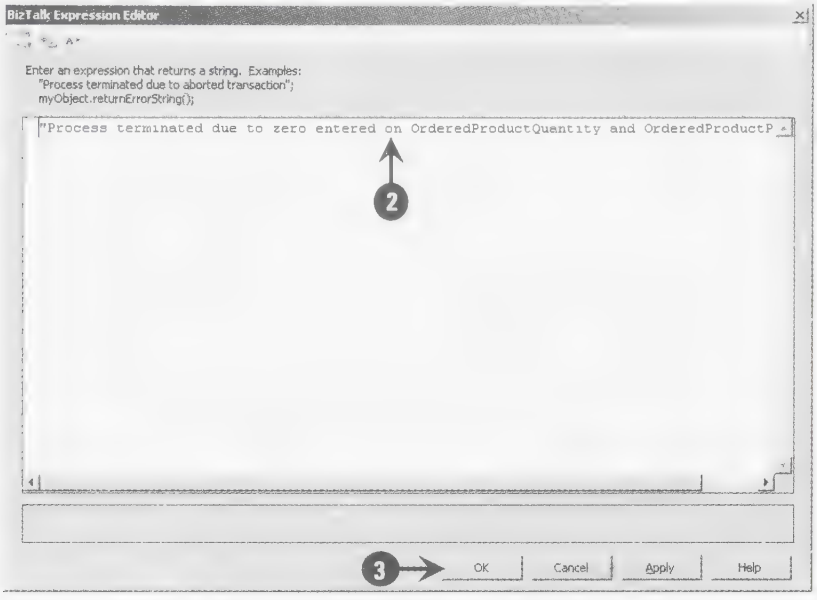


Fig.Biz-4.22

2. Enter the following error message in **BizTalk Expression Editor** dialog box:

```
"Process terminated due to zero entered on OrderedProductQuantity and
OrderedProductPrice"; ( Fig.Biz- 4.22.)
```

3. *Click* the **OK** button to close the **BizTalk Expression Editor** dialog box.

Now the schema based project **CustomerOrder** is complete and ready to be deployed. However, before deploying the project, you will need to first assign a Strong Name to the project. The steps for assigning a strong name are given in the "Performing the Assignment of Strong Naming to the Assembly" section of Chapter 3.

After assigning a strong name to the assembly, the next task is to deploy the **CustomerOrder** project. We take this up in the next section.

Deploying the Project

After adding the **OrderService** schema, **CustomerOrderMessage**, and different shapes in the **CustomerOrderOrchestration** orchestration to the **CustomerOrder** project, the project is complete and ready to be deployed on BizTalk Server 2006. You can deploy **CustomerOrder** project through the **Microsoft Visual Studio 2005**.

The **CustomerOrder** project can be deployed by the following steps:

1. *Select* **Build→Deploy CustomerOrder** from the **CustomerOrder-Microsoft Visual Studio 2005** screen.

Once you deploy the CustomerOrder project successfully, you will get the successfully deployment with zero (0) error essage at the bottom of the output tab of Visual Studio. When you deploy the CustomerOrder project, the .NET framework will automatically build the project before deploying it.

Once the **CusotmerOrder** project has been successfully deployed on BizTalk Server, it is the time to test the project by accessing it in BizTalk Server 2006. This is discussed next.

Accessing the Application in BizTalk Server 2006

Accessing the **CustomerOrder** project involves configuring the project (developed in Microsoft Visual Studio 2005) for its working in BizTalk Server 2006. You can configure the project through **BizTalk Server Administrator Console** or **BizTalk Explorer**. In Chapter 3, we had configured the BizTalk project through **BizTalk Server Administrator Console** but in this chapter, we will configure the **CustomerOrder** project through **BizTalk Explorer**.

You must deploy the CustomerOrder project before accessing it. To know how to deploy CustomerOrder, refer to the Deploying the Project section of this chapter.

Using the BizTalk Explorer

The BizTalk Explorer tool is used for configuring the BizTalk project. Configuring the **CustomerOrder** project in **BizTalk Explorer** involves the following steps:

- Adding and Configuring Receive Port
- Adding and Configuring Send Port
- Configuring Orchestration

We will discuss the steps shortly this chapter. These steps have to be performed within **BizTalk Explorer**. Therefore, let's first open **BizTalk Explorer** in Microsoft Visual Studio 2005 by following these steps:

1. *Select* **View→BizTalk Explorer** to open the **BizTalk Explorer** dialog box, as shown in Fig.Biz-4.23.

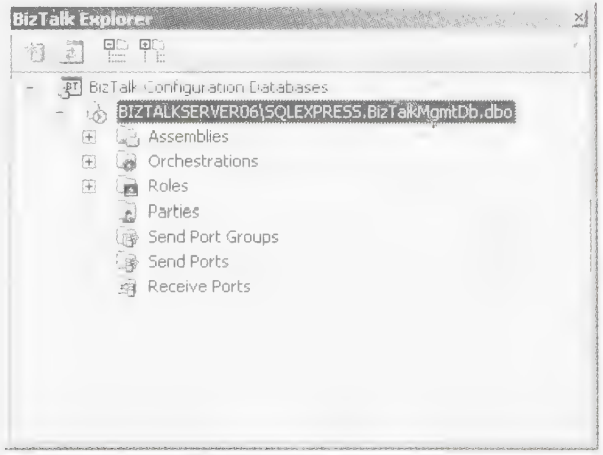


Fig.Biz-4.23

2. *Expand* the **BIZTALKSERVER06\SQLEXPRESS.BizTalkMgmtDb.dbo** node from Fig.Biz-4.23 to add receive ports in the Receive Ports folder, send ports in the Send Ports folder, and configure orchestrations in the Orchestration folder.

Now you need to add physical send/receive ports and connect them to the logical send/receive ports (**CustomerOrderReceivePort** and **CustomerOrderSendPort** in our case) that are added in **CustomerOrderOrchestration** to exchange XML schema messages.

Adding and Configuring Receive Port

BizTalk Explorer is used to connect the logical receive ports with the physical receive ports. You need to bind these two ports in order to exchange XML messages among the different computers within an organization. We have already configured the logical receive ports in the 'Adding send/receive shapes' section of this chapter. The Physical Receive Ports can be added and configured by following these steps:

1. *Right-click* the **Receive Ports** node and *Select* the **Add Receive Ports** option from the context menu to open the **Create New Receive Port** dialog box, as shown in Fig.Biz-4.24.
2. *Select* the type of Receive Port as **One-Way Port** from the drop-down list (Fig.Biz-4.24).

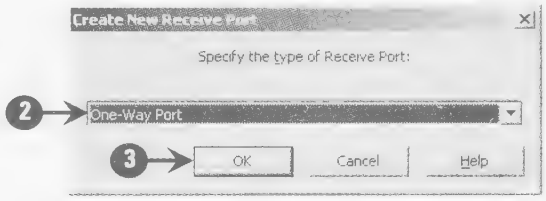


Fig.Biz-4.24

3. *Click* the **OK** button to open the **One-Way Receive Port** dialog box, as shown in Fig.Biz-4.25.
4. *Enter* the name of the Receive Port as **CustomerOrderReceivePort** beside the **Name** field and *click* the **OK** button to create the Receive Port (Fig.Biz-4.25).

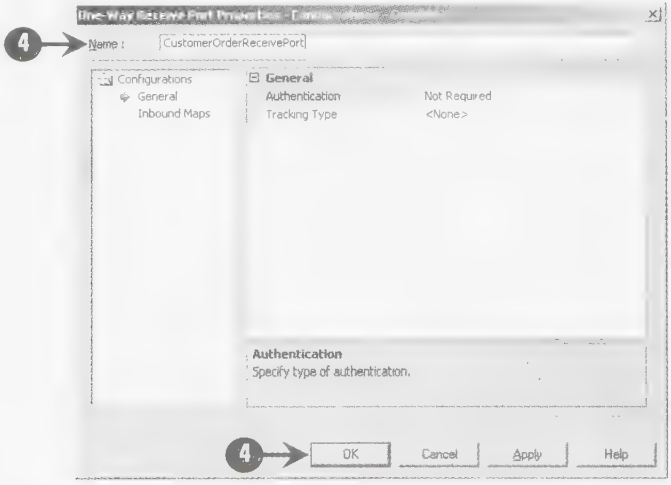


Fig.Biz-4.25

Clicking the **OK** button adds the new Receive Locations to your BizTalk Explorer.

5. *Right-click* **Receive Locations** and *Select* the **Add Receive Location** option from the context menu to add the Receive Location. The **Receive Location** dialog box appears, as shown in Fig.Biz-4.26.
6. *Enter* the values of the following properties on the **One-Way Receive Location Properties** dialog box as shown in Table 4.4, on the **One-Way Receive Location Properties** dialog box.

Table 4.4: Receive location property information

Property	Value
Name	CustomerOrderReceiveLocation
Transport Type	File
Address (URI)	C:\CustomerOrder\receive
Receive Pipeline	Microsoft.BizTalk.DefaultPipelines.XMLReceive (Microsoft.BizTalk.DefaultPipelines.XMLReceive, Microsoft.BizTalk.DefaultPipelines, Version=3.0.1.0,Culture=neutral, PublicKeyToken=31bf3856ad364e35)

When you enter the values against these properties, the resultant **Receive Location** dialog box will appear as shown in Fig.Biz-4.26.

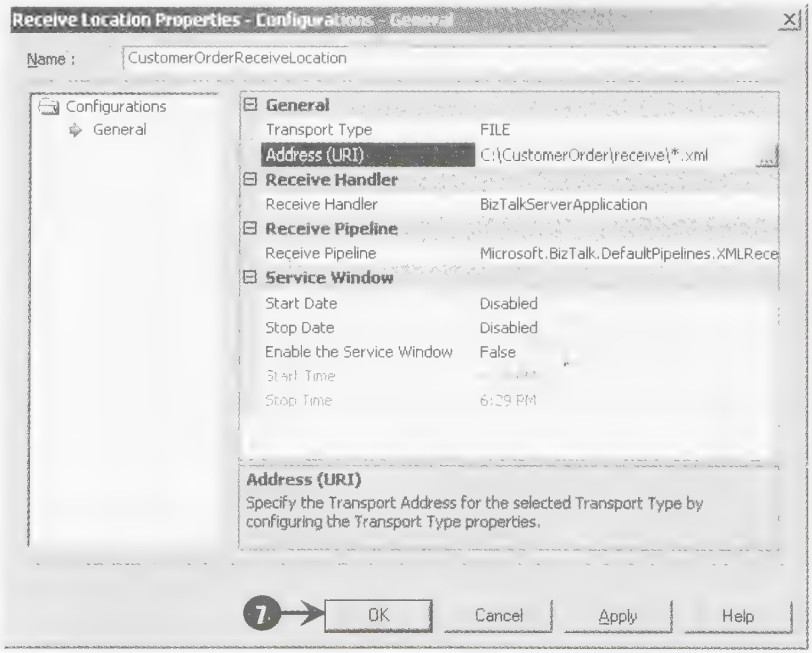


Fig.Biz-4.26

7. Now, *click* the **OK** button to close the **Receive Location Properties** dialog box. **CustomerOrderReceiveLocation** is now added in the **Receive Location** folder of BizTalk Explorer. This is shown encircled in Fig.Biz-4.32.
8. *Right-click* **CustomerOrderReceiveLocation** in BizTalk Explorer and *Select* the **Enable** option from the context menu.

The physical receive port (**CustomerOrderReceivePort**) and its location (**CustomerOrderReceiveLocation**) are now added and configured through BizTalk Explorer, and the next task is to add and configure Send Ports in BizTalk Explorer.

Adding and Configuring Send Port

The Send Port is used to send XML schema messages from the source location to the destination. The Send Port can be added and configured by following these steps:

1. *Right-click* the Send Ports node (Fig.Biz-4.23) and *Select* the **Add Send Port** option from context menu to open the **Create New Send Port** dialog box, as shown in Fig.Biz-4.27.

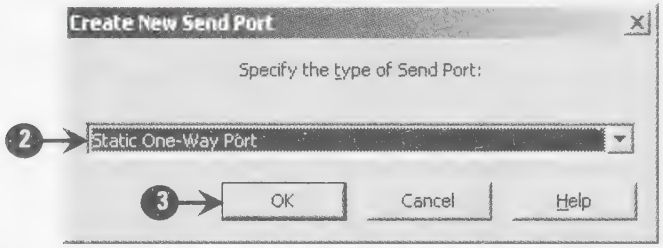


Fig.Biz-4.27

2. *Select* the type of send port as **Static One-Way Port** from the drop down list (Fig.Biz-4.27).
3. *Click* the **OK** button to open the **Static One-Way Send Port** dialog box. as shown in Fig.Biz-4.28.
4. *Enter* the values of the following properties, as shown in Table 4.5 onto the **Static One-Way Send Port Properties** dialog box (Fig.Biz-4.28).

Table 4.5: Send Port property information

Property	Value
Name	CustomerOrderSendPort
Transport Type	File
Address (URI)	C:\CustomerOrder\send\%MessageID%.xml
Send Pipeline	Microsoft.BizTalk.DefaultPipelines.XMLTransmit (Microsoft.BizTalk.DefaultPipelines.XMLTransmit, Microsoft.BizTalk.DefaultPipelines, Version=3.0.1.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35)

To change the values of the properties of the Send Pipeline in Table 4.5, open the Send folder from the left pane of the Static One-Way Send Port Properties dialog box (Fig.Biz-4.28).

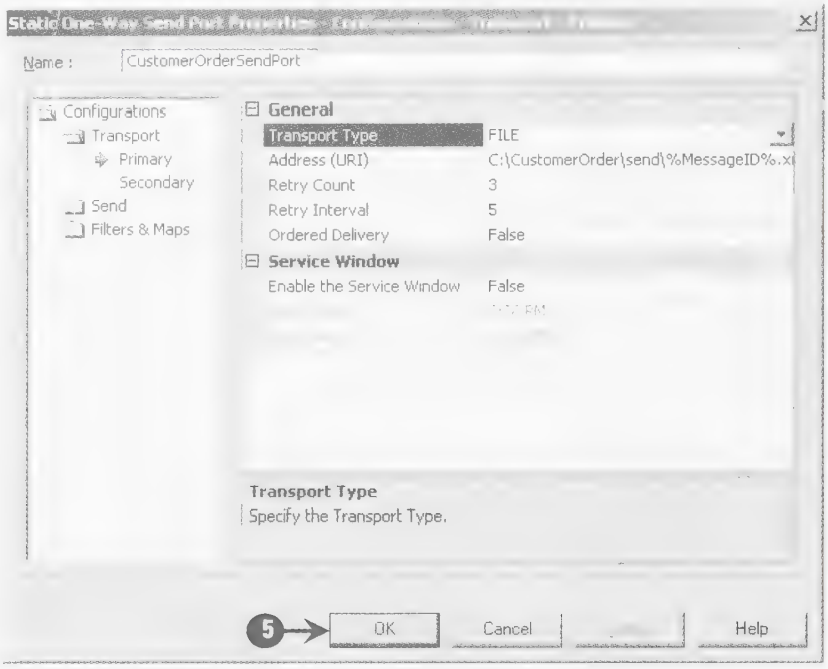


Fig.Biz-4.28

5. *Click* the **OK** button on the **Static One-Way Send Port Properties** dialog box to add the **CustomerOrderSendPort** Send Port in BizTalk Explorer (Fig.Biz-4.28) as well as close the dialog box. This takes you to the **BizTalk Explorer** dialog box (Fig.Biz-4.32).
6. *Right-click* **CustomerOrderSendPort** in the **BizTalk Explorer** dialog box and *Select* the Start option from the context menu to **start** the **CustomerOrderSendPort** Send Port (Fig.Biz-4.32). **CustomerOrderSendPort** is now activated and ready to send schema messages to the destination location.

Now that the Send Port is added onto BizTalk Explorer, the next task is to configure the **CustomerOrderOrchestraton** orchestration.

Configuring Orchestration

The **CustomerOrderOrchestration** orchestration is used to bind the **CustomerOrderReceivePort** (Inbound Ports) with the **CustomerOrderSendPort** (Outbound Ports) in order to send and receive XML schema messages and to assign the BizTalk host name that is used to run the **CustomerOrder** application. These tasks can be performed by the following steps:

1. *Expand* the **Orchestrations** node to open **CustomerOrder.CustomerOrderOrchestration** in **BizTalk Explorer** (Fig.Biz-4.23).
2. *Double-click* **CustomerOrder.CustomerOrderOrchestration** to open the **Port Binding Properties** dialog box, as shown Fig.Biz-4.29.
3. *Select* **CustomerOrderReceivePort** below **Inbound Ports** and **CustomerOrderSendPort** below **Outbound Ports - Static** respectively from the **Port Binding Properties** dialog box (Fig.Biz-4.29).

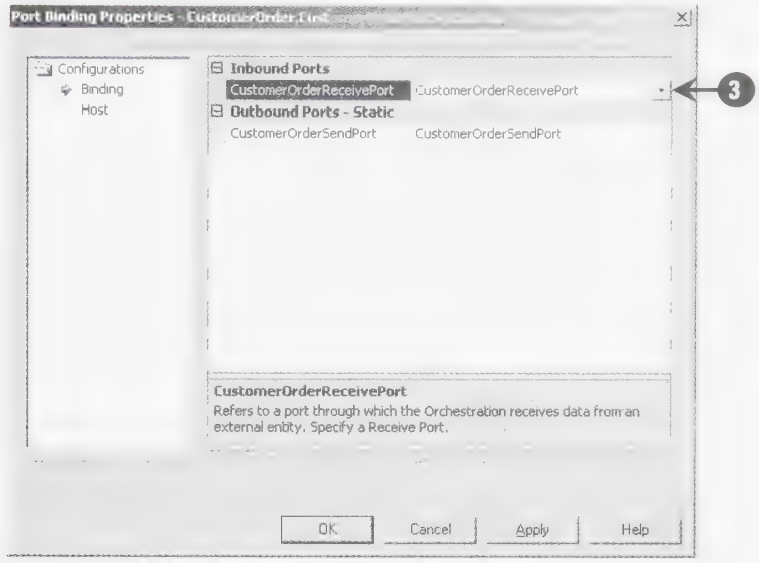


Fig.Biz-4.29

4. Now, *Select* the **Host** node from the **Port Binding Properties** dialog box and *Select* **BizTalkServerApplication** from the **Host** drop-down list, as shown in Fig.Biz-4.30.

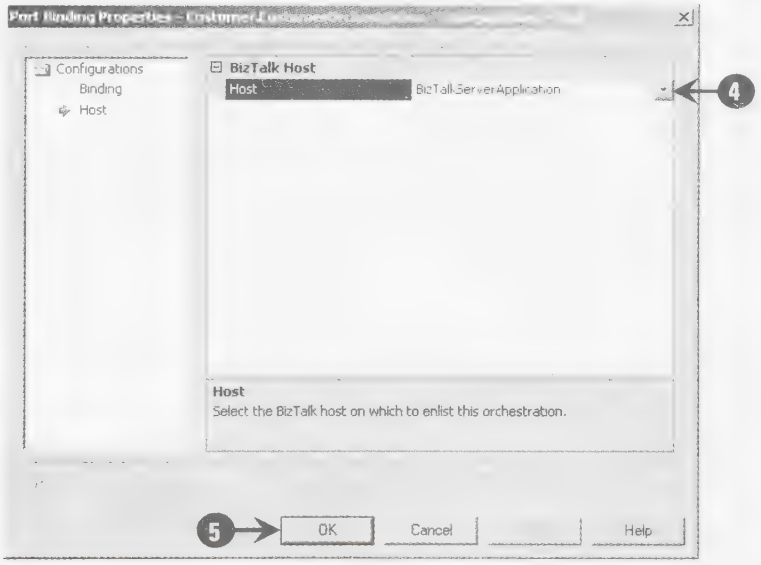


Fig.Biz- 4.30

5. *Click* the **OK** button (Fig.Biz-4.30). The **BizTalk Explorer** dialog box appears on your screen.
6. *Right-click* **CustomerOrder.CustomerOrderOrchestration** (Fig.Biz-4.32) and *Select* the **Start** option from the context menu to start **CustomerOrder.CustomerOrderOrchestration**. This displays the **BizTalk Explorer – Express Start** dialog box, as shown in Fig.Biz-4.31.

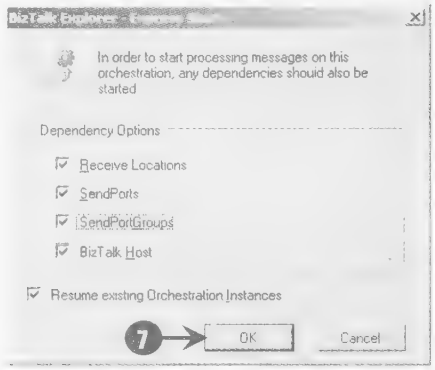


Fig.Biz-4.31

7. *Click* the **OK** button to start **CustomerOrder.CustomerOrderOrchestration**.

After successfully configuring and starting **CustomerOrderReceivePort**, **CustomerOrderReceiveLocation**, **CustomerOrderSendPort**, and **CustomerOrder.CustomerOrderOrchestration**, your BizTalk Explorer will appear as shown in Fig.Biz-4.32.

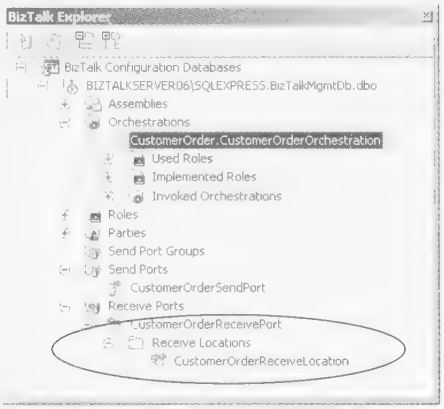


Fig.Biz-4.32

It is now time to test the **CustomerOrder** project.

Testing the Project

You now need to test the **CustomerOrder** project and check whether the **CustomerOrderMessage** message transfers from one location to another on the basis of schema (OrderService).

However, before we go ahead and test the CustomerOrder project, you need to create the **OrderService.xml** file. This file is required to act as a message for transferring message information from one location to another. The **OrderService.xml** file can be obtained by following these steps:

1. *Right-click* **CustomerOrderSchema.xsd** from Solution Explorer (Fig.Biz-4.4) and *Select* the **Generate Instance** option from the context menu to generate the **OrderService.xml** file.

A successful creation of OrderService.xml file message is shown in the Output pane of Microsoft Visual Studio 2005 as soon as the file is created.

After creating **OrderService**.xml file, it is time to test of the **CustomerOrder** project. This can be done by the following steps:

1. Copy the **OrderService.xml** XML file to the **receive** folder location, that is **C:\CustomerOrder\receive**, and wait for BizTalk Server to send that file to the destination folder (**C:\CustomerOrder\send**), as shown in Fig.Biz-4.33.

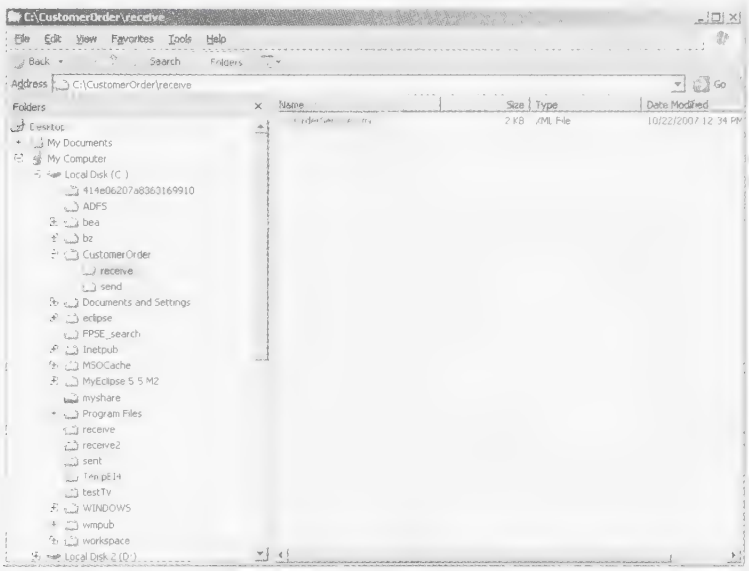


Fig.Biz-4.33

2. After a few second, the **OrderService.xml** file is transferred to the destination folder, that is **C:\CustomerOrder\send**. To check the **OrderService.xml** file, type the following URL: C:\CustomerOrder\send\{18C4F661-C9F7-4E39-B508-3DA547B6DEEE}.xml in Internet Explorer, as shown in Fig.Biz-4.34.

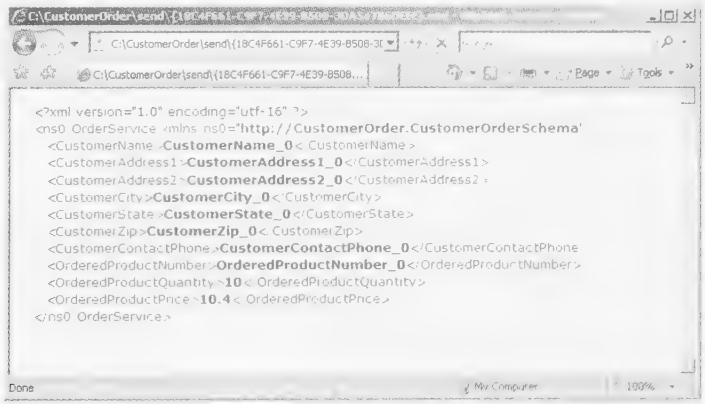


Fig.Biz-4.34

When you transfer the **OrderService.xml** file to the destination folder (C:\CustomerOrder\send), the name of the OrderService.xml file changes in the destination location. In our case, the name changes to {18C4F661-C9F7-4E39-B508-3DA547B6DEEE}.xml file. The name {18C4F661-C9F7-4E39-B508-3DA547B6DEEE}.xml will change again during the testing the application.

This delivery of the **OrderService.xml** file (Fig.Biz-4.34) is an indication that your application is configured properly through BizTalk Explorer and can be easily accessed by BizTalk Server 2006. This ensures that if value of OrderedProductQuantity and OrderedProductPrice is not zero, message transfer process is not terminated.

The **CustomerOrder** project we created is based on BizTalk schema. We are now going to modify the **CustomerOrder** project by adding BizTalk flat file schema to the project. We will also map the contents of incoming messages in flat file format. Let us thus quickly proceed to learn how to develop a BizTalk flat file schema in the **CustomerOrder** project.

Working with Flat File Schemas

A flat file is a computer file that can only be read or written sequentially. It consists of one or more records separated by a comma, tab, pipe, or just plain text in a single string. Each record contains one or more field instances and each field instance can contain a data value.

A flat file contains machine-readable data that is typically encoded as printable characters and usually contains a series of records (or lines), where each record is a sequence of fields. Flat files are needed in BizTalk Server to convert flat file schemas to XML schemas, which can then be added to messages to be sent or received from one location to another. Let's take the example of a simple flat file containing employee data. The file contains one or more employee records and each record contains the following three fields:

- Employee's social security number (ssn)
- Employee's name
- Employee's salary

The contents of the employees' flat file are as follows:

- 123456789,"Mahtab",100000.00
- 444556666,"Santosh ", 87000.00
- 777227878,"Ambrish ",123000.00

Each record contains information of a single employee. The format of the flat file is Comma Separated Value (CSV), which means that each field is terminated by a comma.

Now, the flat file of the employees' record can be converted into XML schema by the following XML document:

```
<?xml version='1.0'?>
<employees>
    <employee>
        <ssn>123456789</ssn>
        <name>Mahtab</name>
        <salary>100000.00</salary>
    </employee>
    <employee>
```

```
            <ssn>444556666</ssn>
            <name>Santosh</name>
            <salary>87000.00</salary>
        </employee>
        <employee>
            <ssn>777227878</ssn>
            <name>Ambrish</name>
            <salary>123000.00</salary>
        </employee>
    </employees>
```

Let us now learn how to add a flat file schema to our CustomerOrder project.

Adding a Flat-File Schema to the Project

Flat file schemas are used to define the structure of the same record and field characteristics as defined in XML schemas format. It also lets you translate a flat file instance message into an equivalent XML instance message (or vice versa). A flat file schema can be added in the **CustomerOrder** project by following these steps:

1. *Open* the **CustomerOrder** project.
2. *Right-click* **CustomerOrder** from the Solution Explorer pane and *Select* **Add→New Item** from the context menu to open the **Add New Item – CustomerOrder** dialog box, as shown in Fig.Biz-4.35.
3. *Select* **Schema Files** from the **Categories** pane and **Flat File Schema** from the **Templates** pane (Fig.Biz-4.35).
4. *Enter* the name of flat file schema as **CustomerOrderFlatFile.xsd** beside the **Name** field (Fig.Biz-4.35)

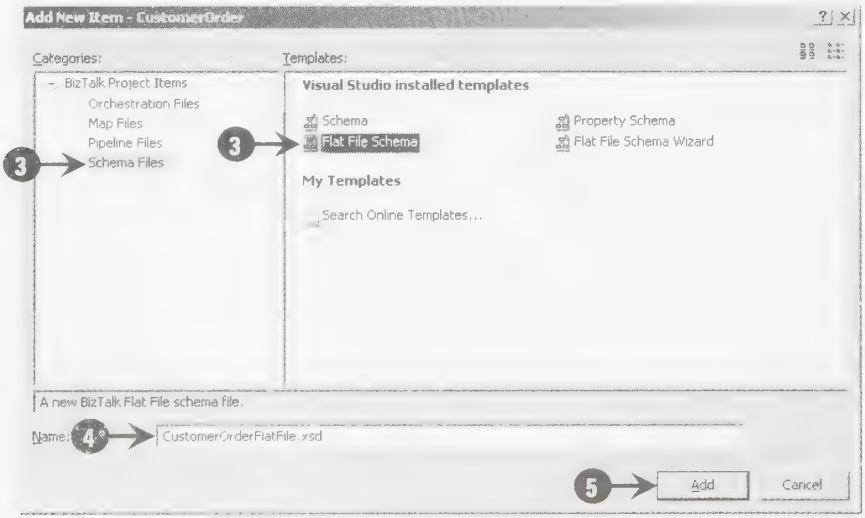


Fig.Biz-4.35

5. Now, *click* the **Add** button to add the flat file schema in the **CustomerOrder** project. A screen as shown in Fig.Biz-4.36 appears.

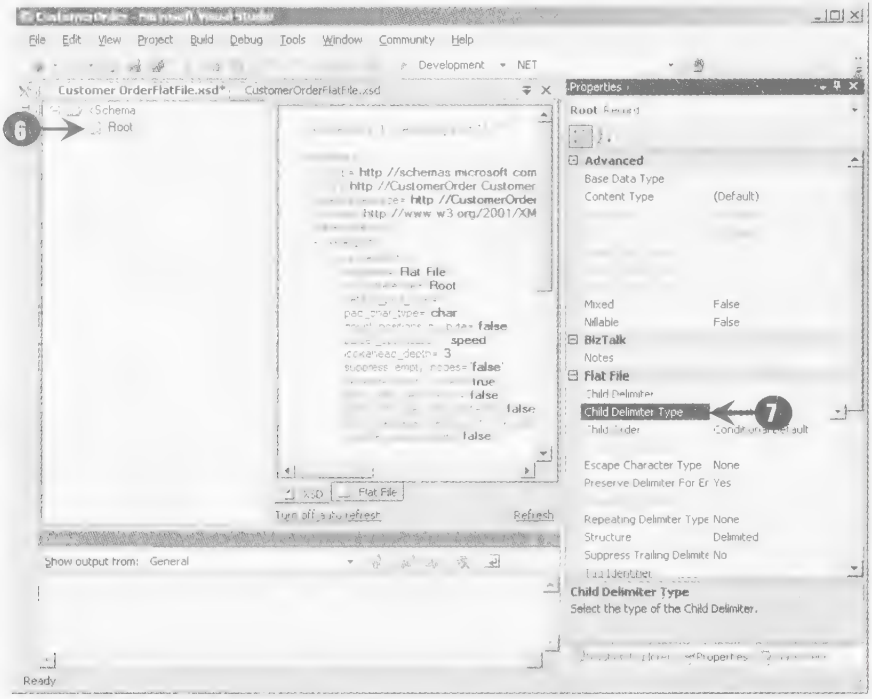


Fig.Biz-4.36

6. *Right-click* the **Root** node and *Select* the **Rename** option from context menu and change the name of the **Root** node to **CustomerOrderService**.
7. *Change* the **Child Delimiter Type** property to **Character** (Fig.Biz-4.36).

The BizTalk **CustomerOrderFlatFile** flat file schema is thus added to the **CustomerOrder** project and the name of **Root** node is changed to CustomerOrderService. The next step is to insert child field elements in the **CustomerOrderService** root node. The child field elements can be added by the following steps:

1. *Right-click* the **CustomerOrderService** node and *Select* **Insert Schema Node→Child Field Element** from the context menu. Enter the name of the child field as CustomerName, shown encircled in Fig.Biz-4.37.
2. *Enter* 40 beside the **Minimum Length with Pad Character** property in the **Properties** pane.

Note

Ensure that all the records that you are creating are of same format. That's why the nodes you are creating are child field element of the root node.

3. Now the **CustomerName** child field is added to the **CustomerOrderService** root node. Repeat step 1 and step 2 to add the following child fields along with their corresponding **Minimum Length with Pad Character** properties, as shown in Table 4.6, to the **CustomerOrderService** root node. The **Minimum Length with Pad Character** property is used to set the length of a field so that it is lesser than or equal to a certain length in your output document.

Table 4.6: Flat File child field name and minimum length	
CustomerAddress1	**40**
CustomerAddress2	**40**
CustomerCity	**30**
CustomerState	**30**
CustomerZip	**10**
CustomerContactPhone	**12**
OrderedProductNumber	**8**
OrderedProductQuantity	**5**
OrderedProductPrice	**10**

After adding all the child fields onto the **CustomerOrderService** root node, you will see a screen as shown in Fig.Biz-4.37.

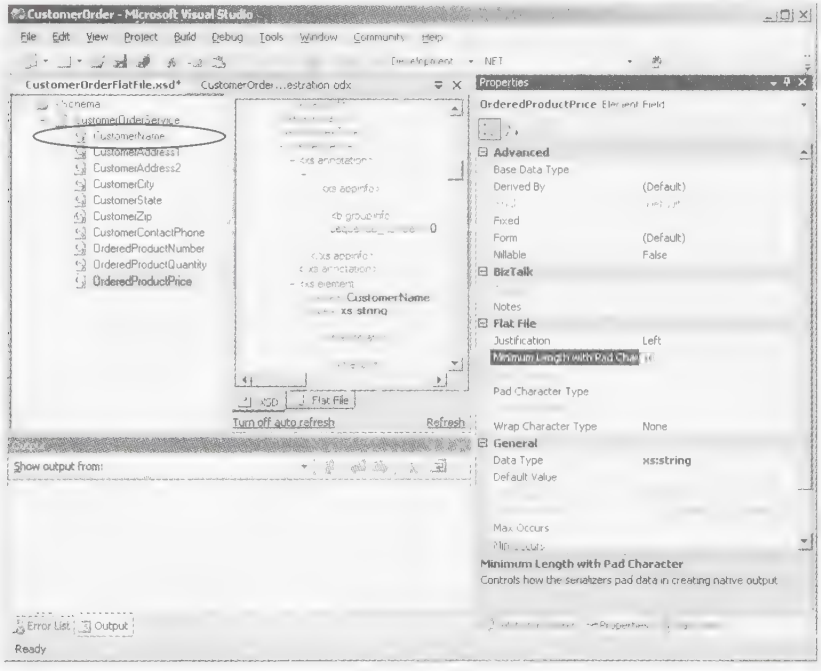


Fig.Biz-4.37

Now save the project. The **CustomerOrderFlatFile** flat file schema has been created successfully. Next, it is time to map the **CustomerOrderMessage** message with the **CustomerOrderFlatFile** flat file schema.

Mapping the Messages

BizTalk Mapper enables you to transform data records of one format (XML schema) to another (flat file schema) and also map the fields in the input record to the fields in the output record. You can use the mapper to map XML messages (instances of XML schema at runtime) to an alternate format (flat file format) to exchange messages. Messages can be mapped by the following steps:

1. *Right-click* **CustomerOrder** from the Solution Explorer pane and *Select* **Add→New Item** from the context menu to open the **Add New Item – CustomerOrder** dialog box, as shown in Fig.Biz-4.38.
2. *Select* **Map Files** from the **Categories** pane and **Map** from the **Templates** pane (Fig.Biz-4.38).
3. *Enter* the name of map file as **CustomerOrderMap.btm** beside the **Name** field (Fig.Biz-4.38).

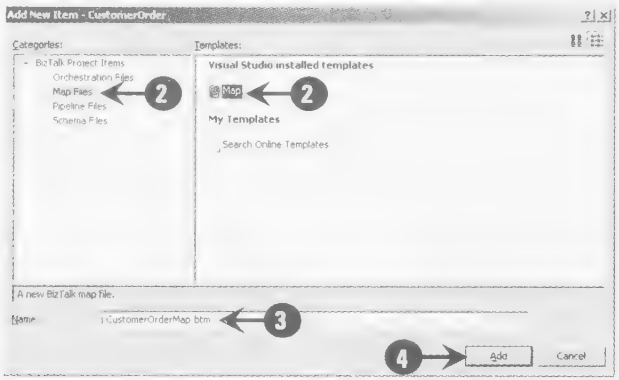


Fig.Biz-4.38

4. Now, *click* the **Add** button to add the map file. A screen similar to the one shown in Fig.Biz-4.39 appears.

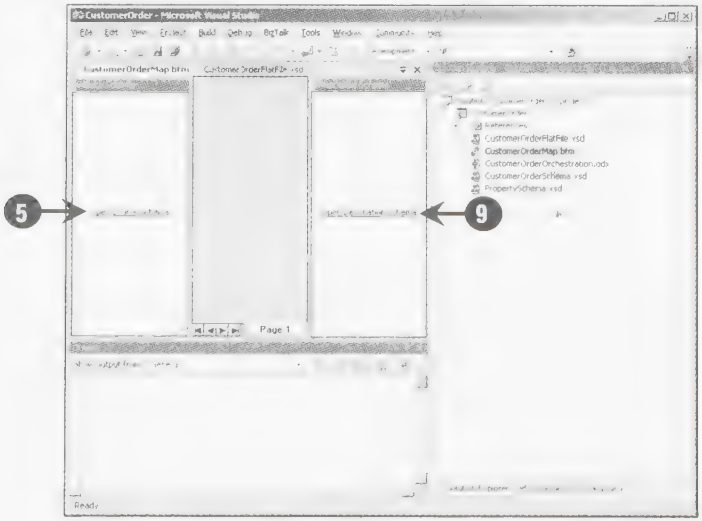


Fig.Biz-4.39

5. *Click* the **Open Source Schema** hyperlink (Fig.Biz-4.39). This will display the **BizTalk Type Picker** dialog box, as shown in Fig.Biz-4.40.

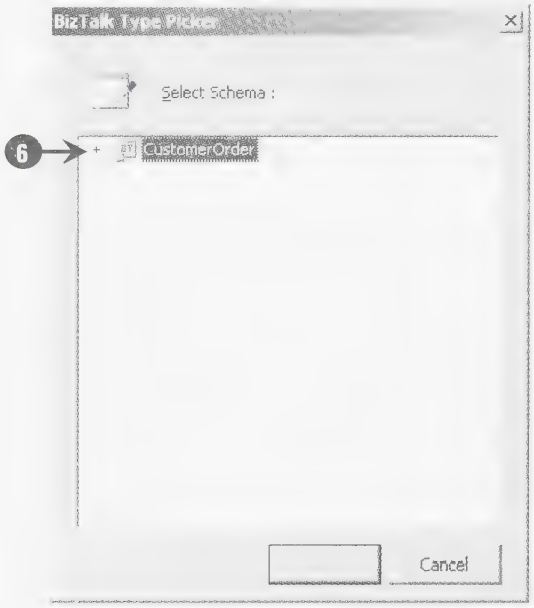


Fig.Biz-4.40

6. Expand the **CustomerOrder** node in the **BizTalk Type Picker** dialog box (Fig.Biz-4.40).
7. Next, expand the **Schemas** node and *Select* **CustomerOrder.CustomerOrderSchema**, as shown in Fig.Biz-4.41.

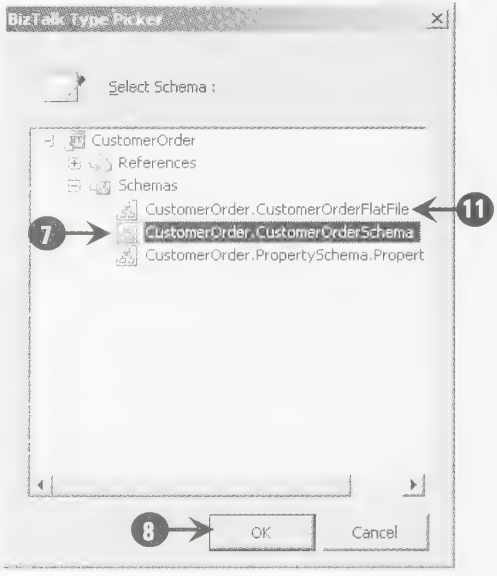


Fig.Biz-4.41

8. Now, *click* the **OK** button to add the **OrderService** schema in **Source Schema**.
9. Similarly, *click* the **Open Destination Schema** hyperlink (Fig.Biz-4.39). The **BizTalk Type Picker** dialog appears (Fig.Biz-4.40).
10. *Expand* the **CustomerOrder** node (Fig.Biz-4.40).
11. Now, expand the **Schemas** node and *Select* **CustomerOrder.CustomerOrderFlatFile** (Fig.Biz-4.41).
12. *Click* the **OK** button to add the **CustomerOrderService** schema in **Destination Schema**.
13. Next, expand the **OrderService** and the **CustomerOrderService** nodes. You will see a screen as shown in Fig.Biz-4.42
14. Connect the child field names in **Source Schema** to their corresponding child field names in **Destination Schema** by dragging each child field in **Source Schema** and dropping it in Destination Schema (for example, drag the **CustomerName** child field in **Source Schema** and drop it in the **CustomerName** child field in **Destination Schema**.
15. Similarly, connect all the child fields of **Source Schema** with their corresponding child fields in **Destination Schema**, as shown in Fig.Biz-4.42.

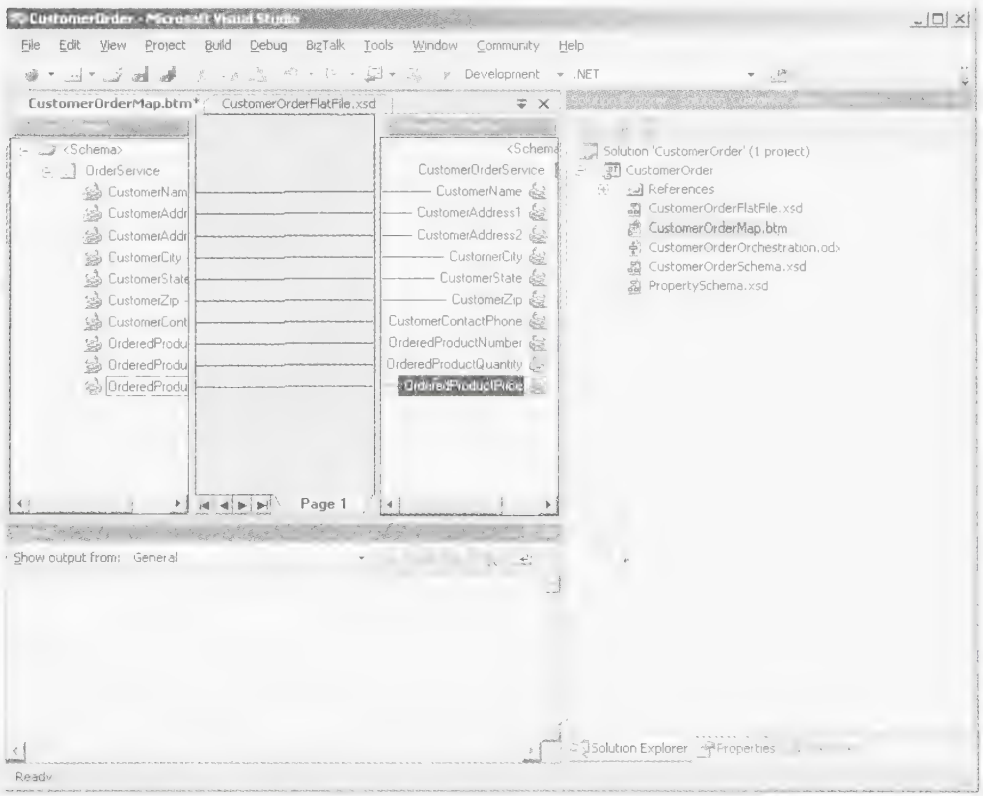


Fig.Biz-4.42

After we have mapped the child fields in **Source Schema** to those in **Destination Schema**, the next step is to validate the mapping to ensure that it has been done in the correct format.

Validating the Mapping

You need to validate the mapping for internal inconsistencies, or other errors that may prevent it from being used effectively for mapping schemas. You can validate the mapping by the following step:

1. *Right-click* **CustomerOrderMap** from the Solution Explorer pane and *Select* the **Validate Map** option from the context menu to validate the map. The following message will appear in the Output window:

```
Invoking component...
C:\Documents and Settings\Administrator\My Documents\Visual Studio
2005\Projects\CustomerOrder\CustomerOrder\CustomerOrderMap.btm: The output XSLT
is stored in the following file: <file:///C:\Documents and
Settings\Administrator\Local Settings\Temp\_MapData\CustomerOrderMap.xsl>
C:\Documents and Settings\Administrator\My Documents\Visual Studio
2005\Projects\CustomerOrder\CustomerOrder\CustomerOrderMap.btm: The Extension
Object XML is stored in the following file: <file:///C:\Documents and
Settings\Administrator\LocalSettings\Temp\_MapData\CustomerOrderMap_extxml.xml>
Component invocation succeeded.
```

This message shows that the mapping of **CustomerOrderMap** has been successfully validated. The next task is to test the map, which we cover in the following section.

Testing the Mapping

A mapping is tested to verify whether the map you designed produces the correct output. The testing is used to generate a test instance from the fields in the Source instance document to the Destination instance document. The mapping can be tested by the following step:

1. *Right-click* the **CustomerOrderMap** from Solution Explorer and *Select* the **Test Map** option from the context menu to test the map. The following message will appear in the Output pane:

```
Invoking component...
Test Map used the following file: <file:///C:\Documents and
Settings\Administrator\Local Settings\Temp\inputfile.xml> as input to the map.
Test Map success for map file C:\Documents and Settings\Administrator\My
Documents\VisualStudio2005\Projects\CustomerOrder\CustomerOrder\CustomerOrderMa
p.btm.
The output is stored in the following file: <file:///C:\Documents and
Settings\Administrator\LocalSettings\Temp\_MapData\CustomerOrderMap_output.xml>
Component invocation succeeded.
```

Next, let's add a flat file schema based message that will be used to exchange information among different locations.

Adding a Schema to Messages

In this project, we use the **CustomerOrderFlatFileMessage** flat file schema as a message, which will be used to exchange information within an organization (which, in our case is a restaurant or hotel). You can add the **CustomerOrderFlatFileMessage** schema onto the Message folder by following these steps:

1. *Right-click* the **Messages** folder in the **Orchestration View** pane and *Select* the **New Message** option from the context menu to add a message. The **Message_1** message is added to the **Message** folder.
2. *Change* the properties of the **Message_1** message according to values mentioned in Table 4.7.

Table 4.7: Properties and values of Message_1 message

Property	Value
Identifier	CustomerOrderFlatFileMessage
Message Type	CustomerOrder.CustomerOrderFlatFile (Schemas node)

After applying these property values, **CustomerOrderFlatFileMessage** is added to the **Messages** folder. Let us now add a Port Type that will be used to transfer **CustomerOrderFlatFileMessage**.

Adding a New Port Type to the Orchestration

A Port Type consists of a set of operations such as requests or responses, and the message types that these operations can work on. It also defines the type of messages (XML schema, flat file schema) to be used in the project. A new Port Type can be added by following these steps in **CustomerOrderOrchestration**:

1. *Right-click* the **Port Types** folder from the **Orchestration View** pane and *Select* the **New One-Way Port Type** option from the context menu. **PortType_1** is added to the **Port Type** folder.
2. *Change* the name of **PortType_1** to **CustomerOrderFlatFilePortType** from the Properties pane.
3. *Expand* **CustomerOrderFlatFilePortType** and change the name of **Operation_1** to **CustomerOrderFlatFileOperation** from the **Properties** pane.
4. *Expand* **CustomerOrderFlatFileOperation** and change the **Message Type** property to **CustomerOrder.CustomerOrderFlatFile** from the **Schema** node in the **Properties** pane.

The **CustomerOrderFlatFilePortType** Port Type is now added to **CustomerOrderOrchestration**. Next, let's add different shapes in **CustomerOrderOrchestration** to build the business process of the **CustomerOrder** project.

Adding Different Shapes to the Project

In this section, we add following types of shapes to build the business process of the **CustomerOrder** project:

- Transform Shape
- Port
- Send Shape

Let us start by adding a Transform shape to the project.

Adding a Transform Shape

As discussed in the 'Orchestration Designer Shapes' section of Chapter 2, the Transform shape is used to transform a message from one format to another. Before adding the **Transform** shape, you need to first delete **CustomerOrderTerminateShape** below the **Else** decision box of **CustomerOrderDecideShape** in **CustomerOrderOrchestration** and add the **Transform** shape in its place. In the earlier project of this chapter, we used the **Terminate** shape below the **Else** decision box because we wanted to stop our orchestration and display some error message defined in the BizTalk Expression Editor of the **Terminate** shape. However, in this modified project we are using the **Transform** shape in place of the **Terminate** shape because we don't want to stop our orchestration and instead want to use flat file schema in our modified project. When you add a Transform shape, BizTalk Server automatically adds the Construct Message shape on the upper part of Transform shape.

The Transform shape can be added by the following steps:

1. *Drag* the Transform shape from Toolbox and drop it below the **Else** decision box of **CustomerOrderDecideShape**. This area is shown encircled in Fig.Biz-4.43.
2. *Rename* the Transform shape as **CustomerOrderTransformShape** from the **Properties** pane (Fig.Biz-4.43).
3. *Select* **ConstructMessage_1** and change its name to **CustomerOrderConstructMessage** from the **Properties** pane.

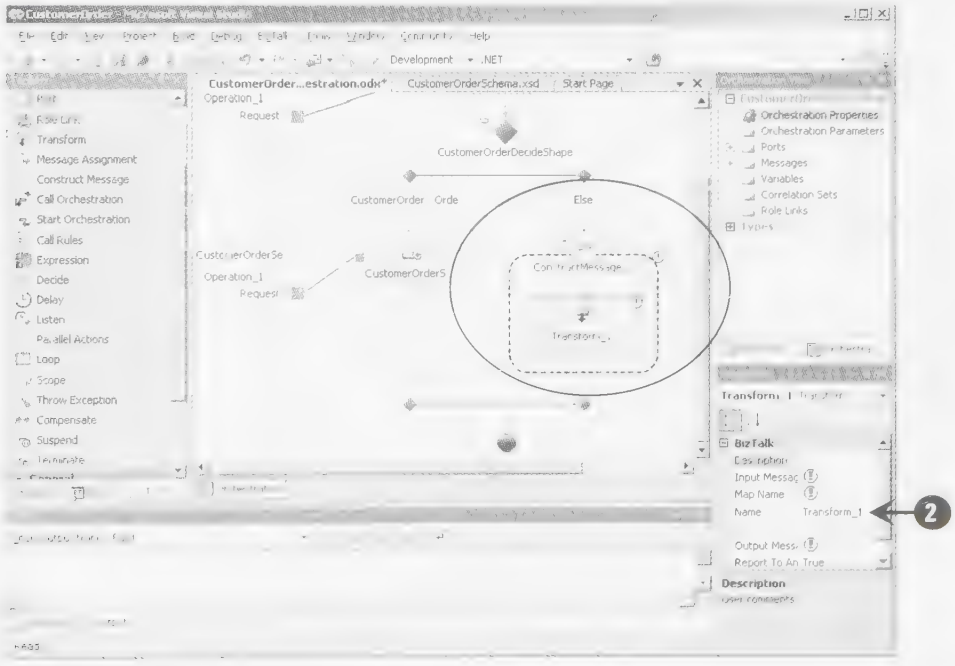


Fig.Biz-4.43

4. *Double-click* **CustomerOrderTransformShape**. The **Transform Configuration** dialog box appears, as shown in Fig.Biz-4.44.

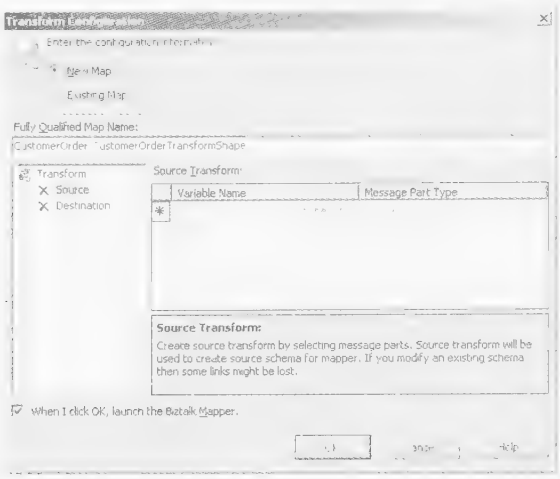


Fig.Biz-4.44

5. *Select* the **Existing Map** radio button in the **Transform Configuration** dialog box (Fig.Biz-.45).
6. *Select* the **CustomerOrder.CustomerOrderMap** option from the **Fully Qualified Map Name** drop-down list (Fig.Biz-4.45).
7. *Select* the **Source** node from the **Transform** pane (Fig.Biz-4.45).
8. *Select* the **CustomerOrderMessage** option from the **Variable Name** drop-down list at the right side of the **Source Transform** pane, as shown in Fig.Biz-4.45

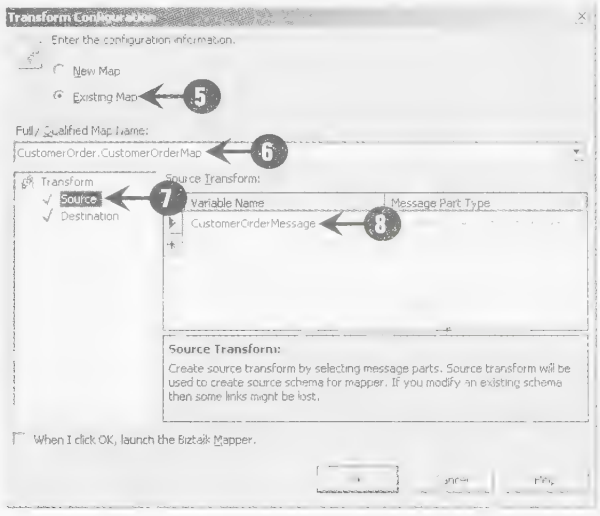


Fig.Biz-4.45

9. *Select* the **Destination** node from the **Transform** pane and *Select* the **CustomerOrderFlatFileMessage** option from the **Variable Name** drop-down list at the right side of the **Destination Transform** pane, as shown in Fig.Biz-4.46.

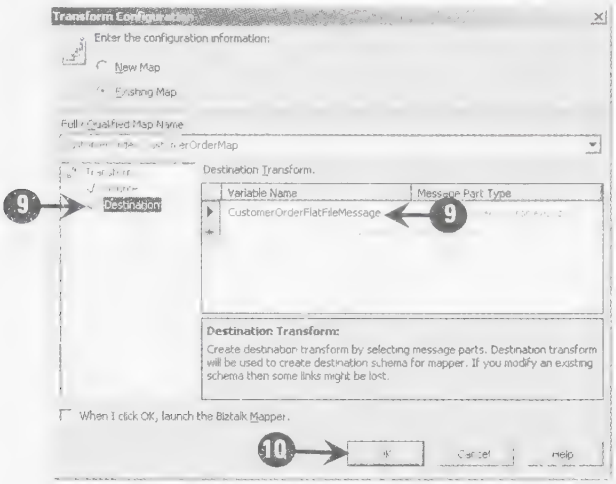


Fig.Biz-4.46

10 *Click* the **OK** button to add the Transform shape in the **CustomerOrderOrchestration** orchestration.

Adding a Port

A Port can be added to the project by the following steps:

1. *Right-click* the **Ports** folder from the Orchestration View pane and *Select* the **New Port** option from the context menu. **Port_1** is added in **CustomerOrderOrchestration**.
2. Next, enter the values of the following properties as shown in table 4.8.

Table 4.8 Properties for Port_1 Port

Property	Value
Identifier	CustomerOrderFaltFilePort
Communication Direction	Send
Port Type	CustomerOrder.CustomerOrderFlatFilePortType (Port Types node).

CustomerOrderFlatFilePort is now added in **CustomerOrderOrchestration**. Let's add a Send shape next.

Adding Send Shape

A Send shape can be added by following these steps in the **CustomerOrderOrchestration** orchestration.

1. *Drag* a **Send** shape from the Toolbox and drop it below the Transform shape in the orchestration surface area. The **Send_1** shape is added to **CustomerOrderOrchestraton**.
2. *Enter* the values of the following properties of **Send_1** in the **Properties** pane as shown in Table 4.9.

Table 4.9 Properties for Send_1 Send Shape	
Property	**Value**
Name	**CustomerOrderFlatFileSendShape**
Message	**CustomerOrderFlatFileMessage**
Operation	**CustomerOrderFaltFilePort.CustomerOrderFlatFileOperation.Request**

3. Save the project and the modified **CustomerOrderOrchestration** will appear as shown Fig.Biz-4.47.

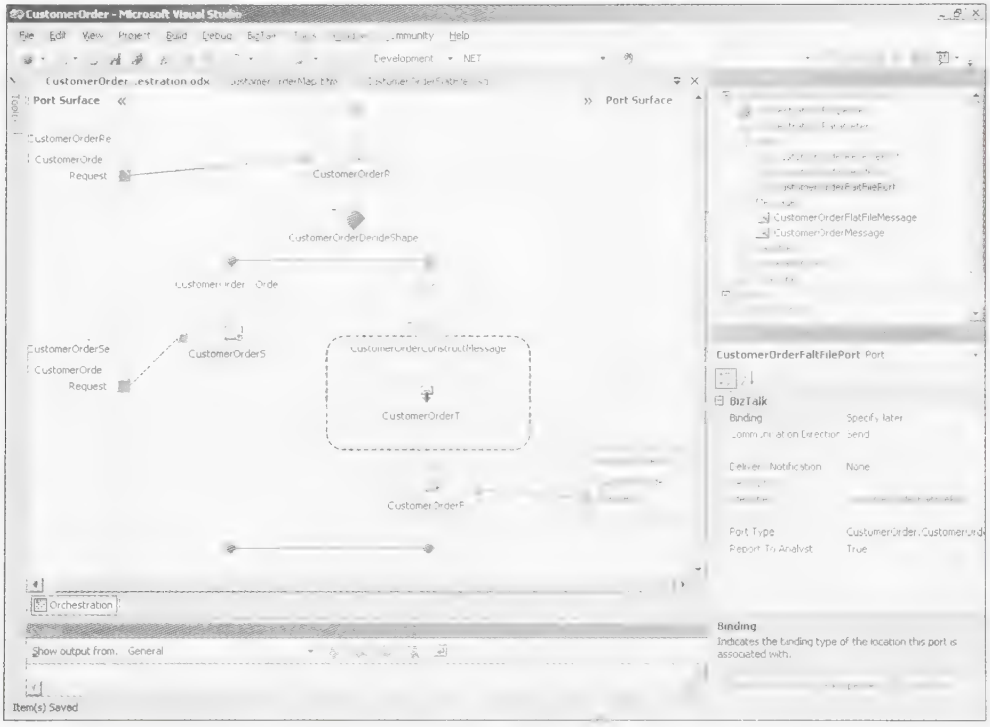


Fig.Biz-4.47

After adding different shapes in **CustomerOrderOrchestration**, the next step is to add a Sen Pipeline through which flat file schema messages can be sent from the source location to th destination location.

Adding a Send Pipeline

Send Pipeline is executed in send port and result will be shown in send location. It is responsibl for processing flat file schema messages before sending them to their final destinations. The Sen pipeline can be added by the following steps:

1. *Right-click* the **CustomerOrder** project from the Solution Explorer pane.
2. *Select* **Add→New Item** from the context menu to add the Send Pipeline. This displays the **Add New Item – CustomerOrder** dialog box, as shown in Fig.Biz-4.48.

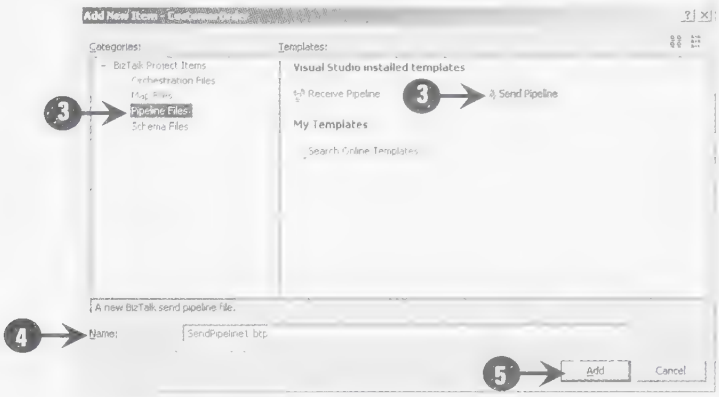


Fig.Biz-4.48

3. *Select* **Pipeline Files** from the **Categories** pane and **Send Pipeline** from the **Templates** pane (Fig.Biz-4.48).
4. *Enter* the name of the **Send** Pipeline as **CustomerOrderSendPipeline.btp** beside the **Name** field (Fig.Biz- 4.48).
5. *Click* the **Add** button to add the Send Pipeline. This displays a screen as shown in Fig.Biz-4.49.

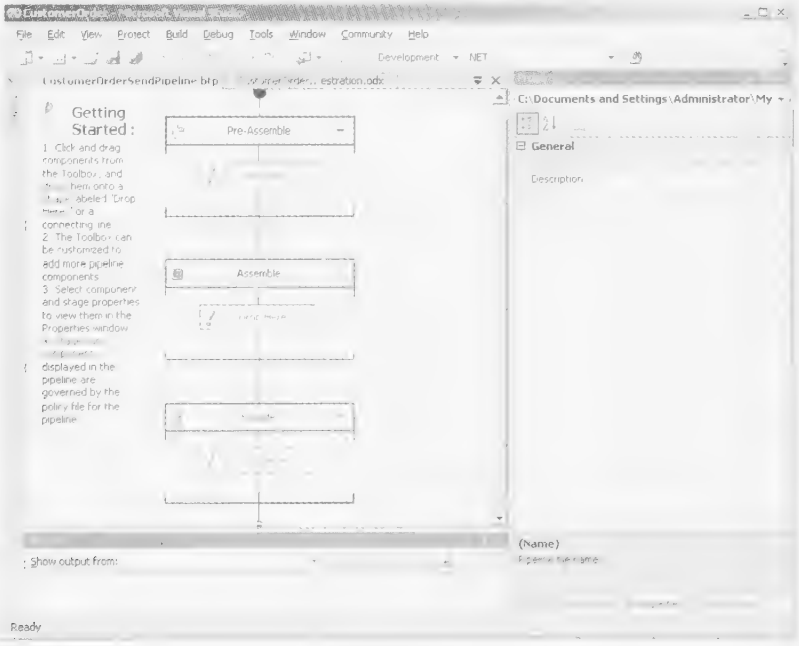


Fig.Biz-4.49

The function of the **CustomerOrderSendPipeline** Pipeline (Fig.Biz-4.49) can be divided into the following three stages:

- **Pre-Assemble:** This stage is used to prepare messages for the outbound process. The customized code for processing the messages is also added in this stage.
- **Assemble:** When BizTalk has finished processing your messages, they are assembled in XML format. In this stage, the data is packed into flat files for shipment to other business entities.
- **Encode:** In this stage, the assembled messages within the Send Pipeline are encoded. In the same way, you can decode the assembled message in the Receive Pipeline.

6. *Drag* a **Flat file assembler** shape from the Toolbox and drop it below the **Assemble** box (Fig.Biz-4.49).
7. *Set* the **Document schema** property to **CustomerOrder.CustomerOrderFlatFile** in the **Properties** pane, as shown in Fig.Biz-4.50.

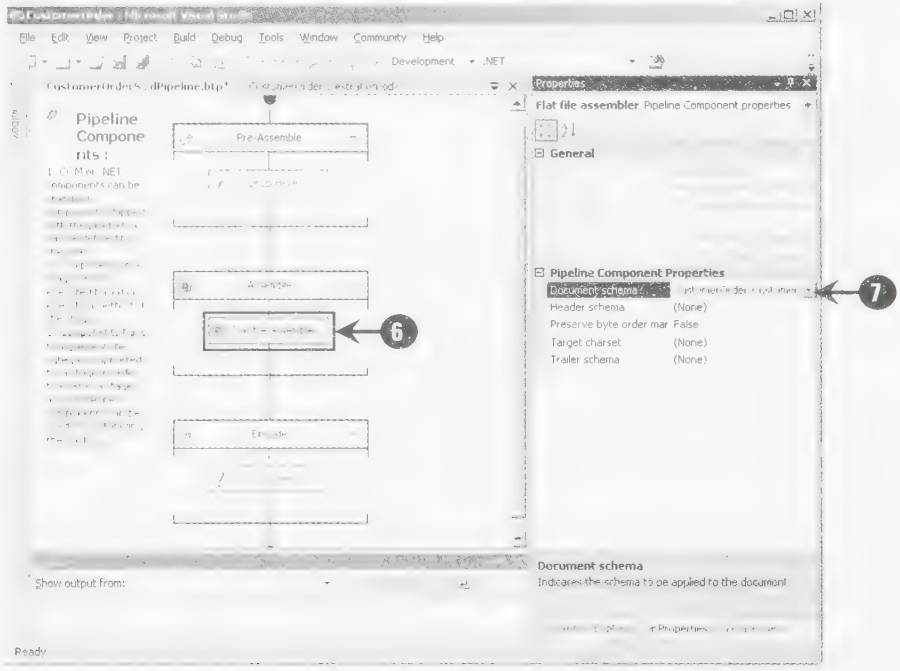


Fig.Biz-4.50

Now the **CustomerOrderSendPipeline** Send Pipeline is added to your **CustomerOrder** project, and with this, the development of the **CustomerOrder** flat file schema project is also complete. Now you need to just save the project and it is ready to be deployed.

Deploying the Project

The modified **CustomerOrder** project can be deployed by the following step:

1. *Select* the **Deploy CustomerOrder** option from the **Build** menu on the **Microsoft Visual Studio 2005** window.

As soon as you deploy your project successfully, you will get the successfully deployment with zero (0) error message at the bottom of the Output tab of Visual Studio.

After the CustomerOrder project is successfully deployed, it is time to access the project. This is described in the next section.

Configuring the Application in BizTalk Server 2006

The modified **CustomerOrder** project can be configured through BizTalk Explorer. This can be done by the following steps:

1. *Select* the **BizTalk Explorer** menu option from the **View** menu in Microsoft Visual Studio 2005 to open **BizTalk Explorer** (Fig.Biz-4.23).
2. *Expand* the **BIZTALKSERVER06\SQLEXPRESS.BizTalkMgmtDb.dbo** node (Fig.Biz-4.23).
3. *Right-click* the **BIZTALKSERVER06\SQLEXPRESS.BizTalkMgmtDb.dbo** node and *Select* the **Refresh** option from the context menu.

We have already added the Receive Port, Send Port and Orchestration while configuring the CustomerOrder project earlier in the chapter. Therefore, here we will add only the Send Port for sending flat file schema messages.

Adding Send Ports

The Send Port can be added by the following steps:

1. *Right-click* the **Send Ports** node (Fig.Biz-4.23) and *Select* the **Add Send Port** option from context menu to open the **Create New Send Port** dialog box (Fig.Biz-4.27).
2. *Select* the **Static One-Way Port** option from the **Create New Send Port** dialog box (Fig.Biz-4.27) and *click* the OK button to open the **Static One-Way Send Port Properties** dialog box, as shown in Fig.Biz-4.28
3. *Enter* the values for the following properties as shown in Table 4.9 in the **Static One-Way Send Port Properties** dialog box (Fig.Biz-4.28).

Table 4.9: Send Port property information

Property	Value
Name	CustomerOrderFlatFileSendPort
Transport Type	File
Address (URI)	C:\CustomerOrder\sendFlat\%MessageID%.txt
Send pipeline	CustomerOrder.CustomerOrderSendPipeline (CustomerOrder.CustomerOrderSendPipeline, CustomerOrder, Version=1.0.0.0, Culture=neutral, PublicKeyToken=fb14a1c1d081e8e4)

To change the values of property of the Send Pipeline, select the Send folder from left pane of the **Static One-Way Send Port Properties** dialog box (Fig.Biz-4.28).

4. *Click* the **OK** button to add the **CustomerOrderFlatFileSendPort** Send Port in the **Static One-Way Send Port Properties** dialog box.
5. Now, *Right-click* **CustomerOrderFlatFileSendPort** and *Select* the **Start** option from the context menu to start the **CustomerOrdeFlatFilerSendPort** Send Port.

Now that the Send Port is added to BizTalk Explorer, the next task is to configure **CustomerOrderOrchestration** through BizTalk Explorer.

Configuring CustomerOrderOrchestration and Selecting Port

We now need to configure **CustomerOrderOrchestration** and *Select* the **CustomerOrderFlatFilePort**, which can be done by the following steps:

1. *Expand* the **Orchestrations** node to open **CustomerOrder.CustomerOrderOrchestration** from **BizTalk Explorer**, as shown in Fig.Biz-4.21.
2. *Double-click* **CustomerOrder.CustomerOrderOrchestration** to open the **Port Binding Properties** dialog box, as shown Fig.Biz-4.51.
3. *Select* the **CustomerOrderFlatFileSendPort** option from the **CustomerOrderFlatFilePort** drop-down list on the **Outbound Port-Static** pane (Fig.Biz-4.51).
4. *Select* the **Host** node. Then *Select* the **BizTalkServerApplication** option from the **Host** drop-down list.
5. Now, *click* the **OK** button to finish the configuration. The BizTalk Explorer dialog box appears.

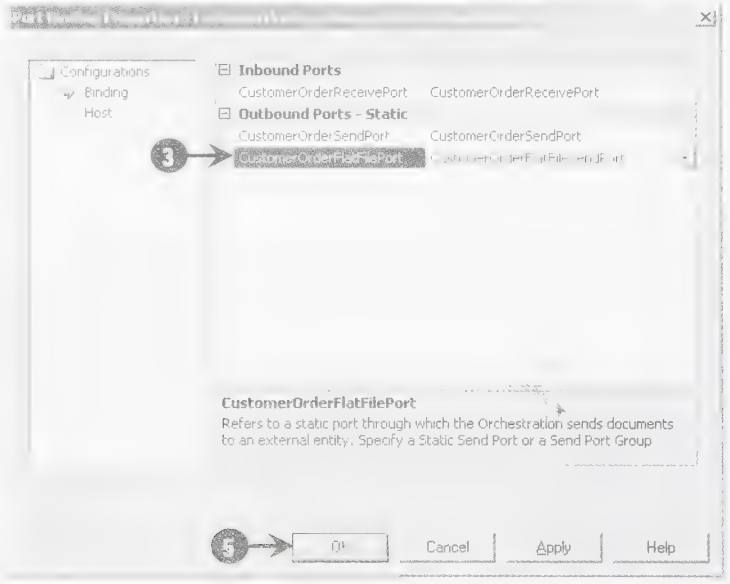


Fig.Biz-4.51

6. *Right-click* the **CustomerOrder.CustomerOrderOrchestration** and *Select* the **Start** option from the context menu to start the **CustomerOrder.CustomerOrderOrchestration** orchestration (Fig.Biz-4.52).

After performing all these steps, the **BizTalk Explorer** will appear as shown in Fig.Biz-4.52.

Fig.Biz-4.52

Enable Receive Location (**CustomerOrderReceiveLocation**) from the **Receive Ports** (**CustomerOrderReceivePort**) node and **CustomerOrderOrchestration**, if it is disabled. Now the **CustomerOrder** project is ready to be tested.

Testing the Application

The **CustomerOrderFlatFile_output.xml** file is needed to test the **CustomerOrder** project. The **CustomerOrderFlatFile_output.xml** file can be obtained by the following step:

1. *Right-click* **CustomerOrderFlatFile.xsd** from **Solution Explorer** and *Select* the **Generate Instance** option from the context menu to generate the **CustomerOrderFlatFile_output.xml** flat file.

As soon as the **CustomerOrderFlatFile_output.xml** file is generated, the following message will appear in the **Output** tab:

```
Invoking component...
Create XML Instance succeeded for schema Customer OrderFlatFile.xsd. Generated
<file:///C:\Documents and Settings\Administrator\Local
Settings\Temp\_SchemaData\CustomerOrderFlatFile_output.xml> as output.
Component invocation succeeded.
```

After creating the **CustomerOrderFlatFile_output.xml** file, open it with notepad editor and enter zero to **OrderedProductQuantity** (<OrderedProductQuantity>0</OrderedProductQuantity>) and also enter zero to **OrderedProductPrice** (<OrderedProductPrice>0</OrderedProductPrice>) child fields. This is require to test the **CustomerOrderFlatFile** schema of **CustomerOrder** project. The project can be tested by the following steps:

1. Copy the **CustomerOrderFlatFile_output.xml** XML file (in our case the location of the **CustomerOrderFlatFile_output.xml** file is **C:\Documents and Settings\Administrator\LocalSettings\Temp_SchemaData\CustomerOrderFlatFile_output.xml**) to the receive folder location (**C:\CustomerOrder\receive**) as shown in Fig.Biz-4.53.

Wait for some time for BizTalk Server to send the XML based flat file to the destination folder (C:\CustomerOrder\send).

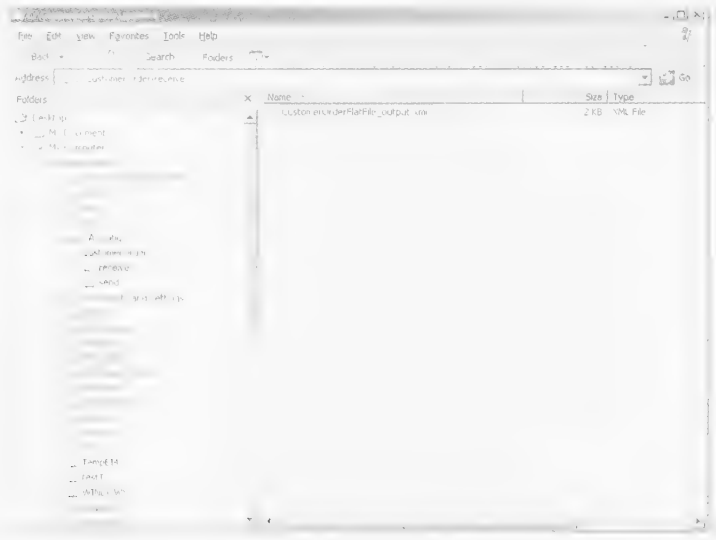


Fig.Biz-4.53

Once the **CustomerOrderFlatFile_output.xml** file is transferred in the destination folder (C:\CustomerOrder\sendFlat), you can check the file {ODD9F629-C445-443F-B7CE-E6D1C65EB58E}.txt in the destination folder C:\CustomerOrder\sendFlat as shown in Fig.Biz-4.54. The file name {ODD9F629-C445-443F-B7CE-E6D1C65EB58E}.txt will change again during the execution of the application. File is converted to TXT format as value of OrderedProductQuantity and OrderedProductPrice is zero, it happens because of the expression defined for **Decide** shape defined earlier.

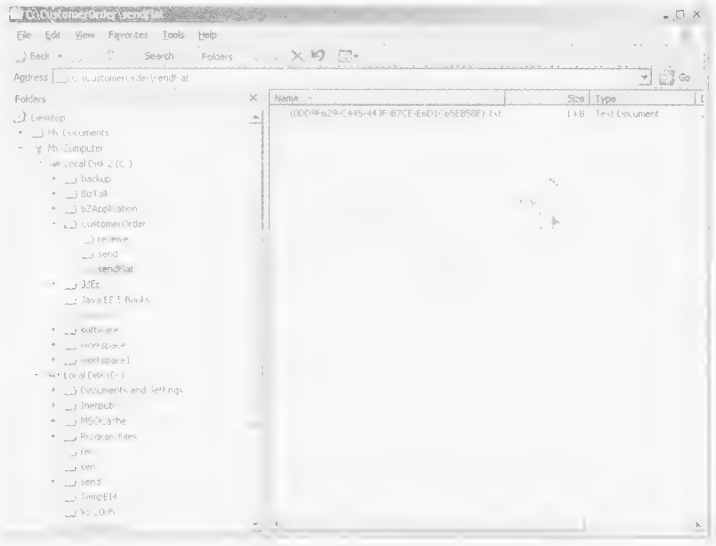


Fig.Biz-4.54

You have deployed and run the flat file schema based project successfully and now are equipped with the knowledge to create schema and flat file based projects of your own. This concludes the chapter. However, before we move to the next chapter, let's recap the main points of this chapter by going through a short summary.

Summary

In this chapter, have learned how to create a schema based BizTalk project in the Microsoft Visual Studio 2005 environment and shown you in simple steps how configure the project through BizTalk Explorer. It takes you through the whole process of adding and configuring the various components needed in the project, after which the project was deployed and tested to check for errors and inconsistencies. After testing the schema based BizTalk project, you learned about the flat file schema by modifying the schema based project. You also learned how to map XML schema base messages to flat file based messages and also configured the modified project through BizTalk Explorer. Finally, you learned how to test the modified flat file schema based project.

In the next chapter, we will discuss business rules and develop a policy by using Business Rule Composer and also call this policy by developing a BZ project.

Chapter 5

Implementing Business Rules

In this Chapter

- Business Rules
- Installing the Business Rule Composer
- Creating a Policy
- Calling the Policy from an Orchestration

In BZ 2006, business rules or business policies are used to define and control the structure, operations, constraints, and strategies of an organization. They are dynamic and subject to change over time, and can be found in all types of business applications. Finance and insurance, e-business, transportation, telecommunications are few of the example that are governed by business rules.

This chapter introduces BizTalk business rules and policies in detail. We will also learn about the Business Rule Composer and how we can install it. We will also create, publish, and test a policy by using the Business Rule Composer. In the end, we will discuss how to call this policy in our BizTalk project.

Business Rules

Business rules are agreements, statements and contracts that govern the conduct of business processes. A business rule defines as a rule that contains one aspect of your business that is intended to assert business structure or influence the behavior of your business. Business rules may focus on business calculations. For example, the record of a purchase order may not be entered if the customer's credit is less than 50. Some business rules focus on the policies of an organization. An example of business policy in a car rental company is "only legal and good condition car will be rented to customer". A business rule consists of three parts:

- **Rule**: Rules are declarative statements that govern business processes. Rule determines the business logic, in the form of a comparison of two monetary values, to data or facts, in the form of a transaction amount and available funds. You can use the Business Rule Composer to create, modify, version, and deploy the business rules. Rules are defined by following format: `IF condition THEN action`, which consists following parts:
  - **Condition**: A condition is Boolean expression that consists of one or more predicates and facts. Predicates are "is less than", "equal to", "is greater than" and so on. A predicates can also combined with logical connectives such as AND, OR and NOT.
  - **Actions**: Actions are the functional consequences of condition evaluation. If a rule condition is met, the corresponding action or actions of the condition are preformed.
  - **Facts**: Facts are the data on which business rules operate. For example, "amount" and "available funds" are facts.

To make the picture clearer about condition, action and facts, let us see an example here. Consider the following statement:

IF amount is less than or equal to available funds THEN conduct transaction and print receipt.

In this example, condition consists two parts:

"is less than or equal to"

"amount" and "available fund"

Actions consist of events that are executed, such as "conduct transaction" and "print receipt", when the condition evaluates to true. Facts as discussed earlier are data such as "amount" and "available funds".

- **Policy:** A policy consists of rules in a group. You need to compose a policy version and test the policy using the facts defined in the rules. Policies can be composed using the Business

Rule Composer. While using the Business Rule Composer, you need to mention the rules that are related to these polices.

- **Vocabulary:** A vocabulary is a collection of definitions, which includes common names of the facts used in conditions and actions defined for the rules. Vocabulary helps in understanding the rules clearly as vocabulary definitions make the rules easier to read. You can build vocabulary through Business Rule Composer.

You need to ensure that the policy for a rule is defined before using its vocabulary. This helps in allotting a version number to the vocabulary. In addition, vocabulary needs to be published through the Business Rule Composer. A published vocabulary is provided for preserving its referential integrity and for restricting modifications in the vocabulary.

Let's now learn how to create a policy and rule in BZ 2006 by using the Business Rule Composer.

Installing the Business Rule Composer

Business Rule Composer (BRC) is a development tool, which is provided by BZ 2006 for authoring, versioning, creating and deploying the policies and vocabularies of your business domain. Before you can use the BRC, you need to first install it in your computer. This can be done by using the following steps:

1. *Insert* the BZ 2006 CD/DVD in the CD-ROM. The CD will start auto-run menu screen as shown in Fig.Biz-5.1.

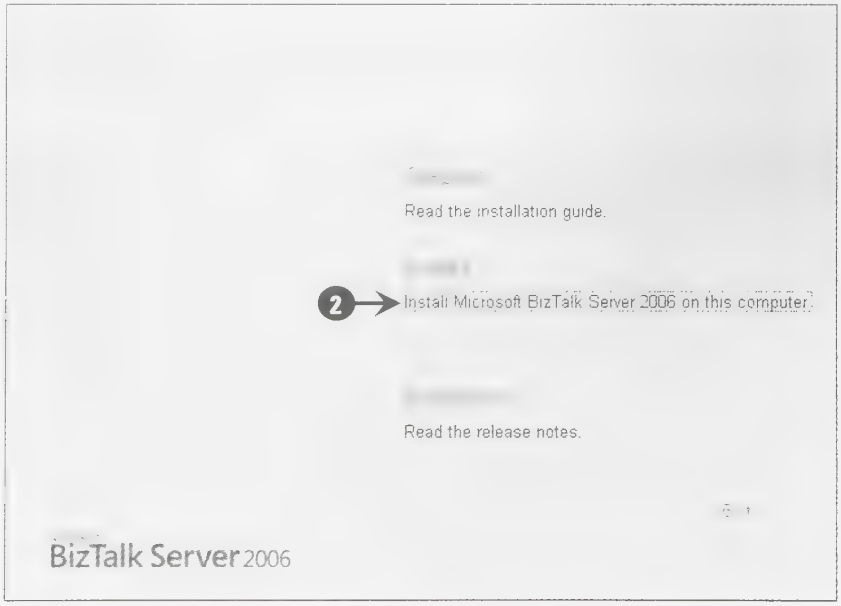


Fig.Biz-5.1

2. *Click* the **Install Microsoft BizTalk Server 2006 on this computer** link (Fig.Biz-5.1) to open the **Program Maintenance** screen, as shown in Fig.Biz-5.2.

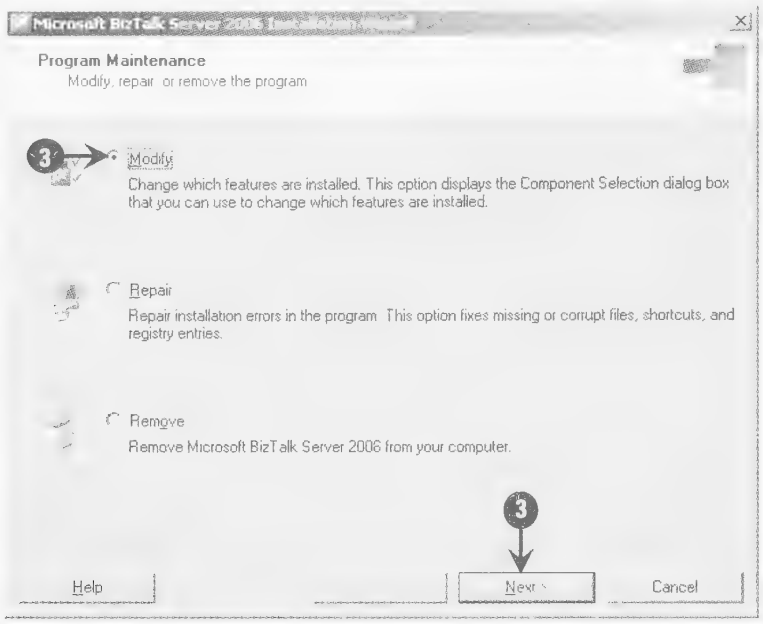


Fig.Biz-5.2

3. *Select* the **Modify** radio button (Fig.Biz-5.2) and *click* the **Next** button to open the **Component Installation** screen, as shown in Fig.Biz-5.3.

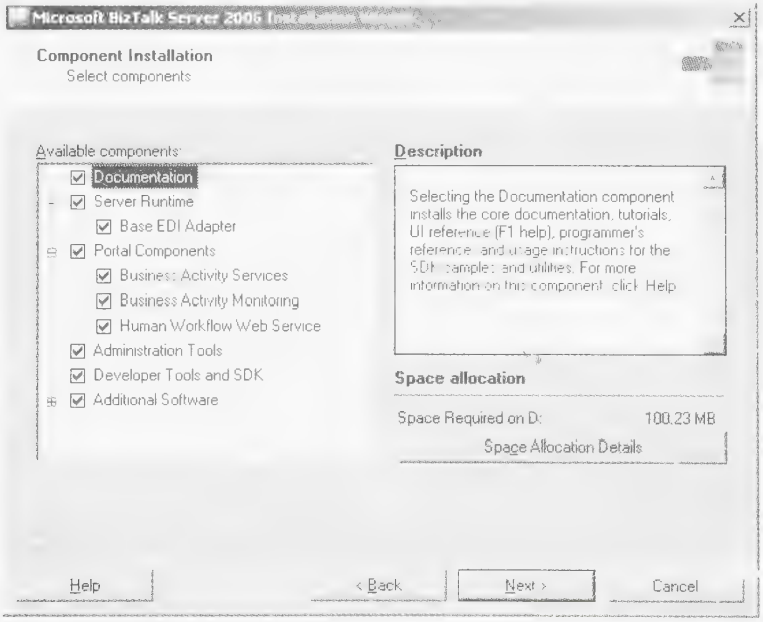


Fig.Biz-5.3

4. *Expand* the **Additional Software** node (Fig.Biz-5.3) and *select* the **Business Rules Components** subnode on the **Component installation** screen, as shown in Fig.Biz-5.4.

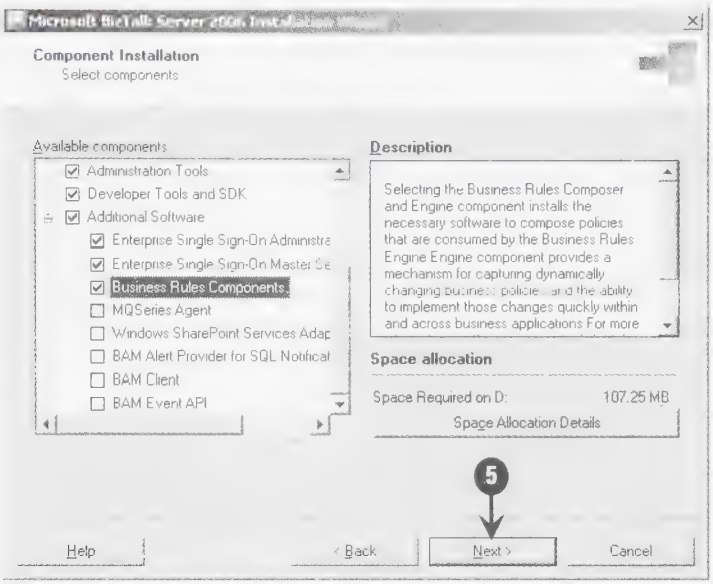


Fig.Biz-5.4

5. *Click* the **Next** button (Fig.Biz-5.4) to open the **Summary** screen, as shown in Fig.Biz-5.5.

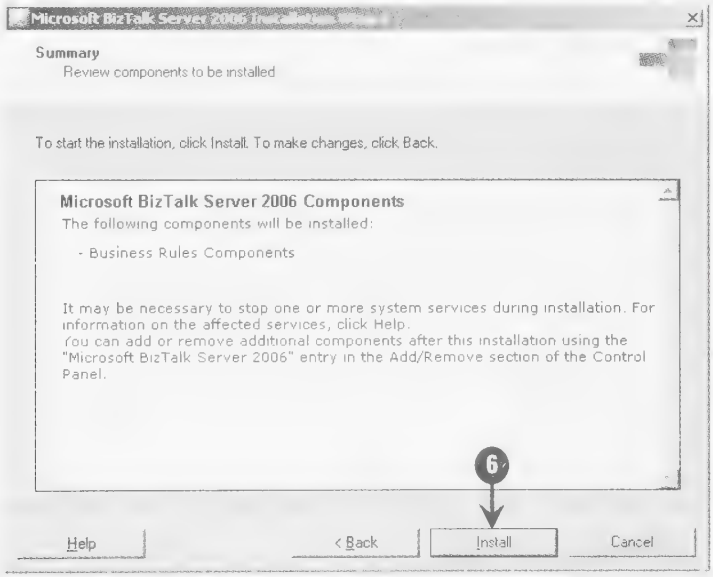


Fig.Biz-5.5

In the **Summary** screen, you can view the components that we are going to install.

6. *Click* the **Install** button on the **Summary** screen to start the installation process, as shown in Fig.Biz-5.6.

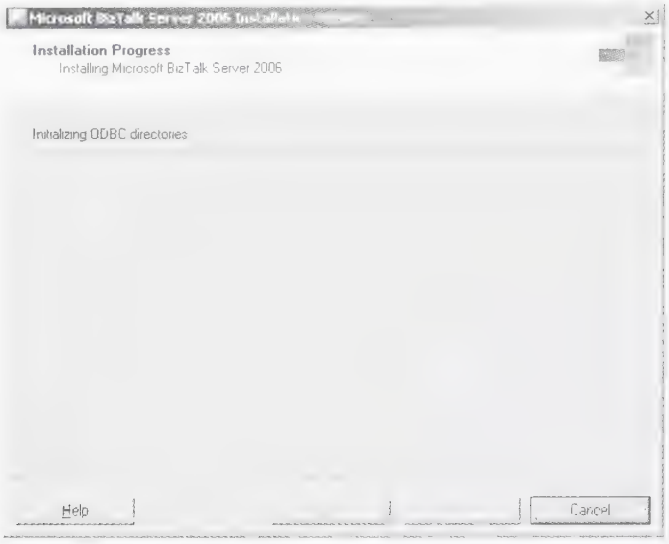


Fig.Biz-5.6

After the installation process gets completed, the **Installation Completed** screen opens, as shown in Fig.Biz-5.7.

7. *Click* **Finish** to close the screen and open the **BizTalk Server Configuration** screen. Now, you can configure the BizTalk Server as mentioned in Chapter 1 of the book.

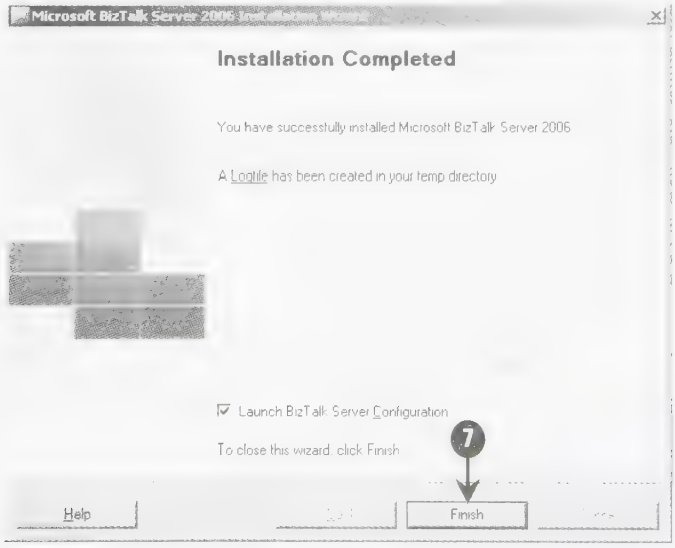


Fig.Biz-5.7

Now that the BizTalk Rule Composer (BRC) is installed in your computer, let's access it by the following steps:

1. *Click* **Start→All Programs→Microsoft BizTalk Server 2006→Business Rule Composer** to open the BRC screen, as shown in Fig.Biz-5.8.

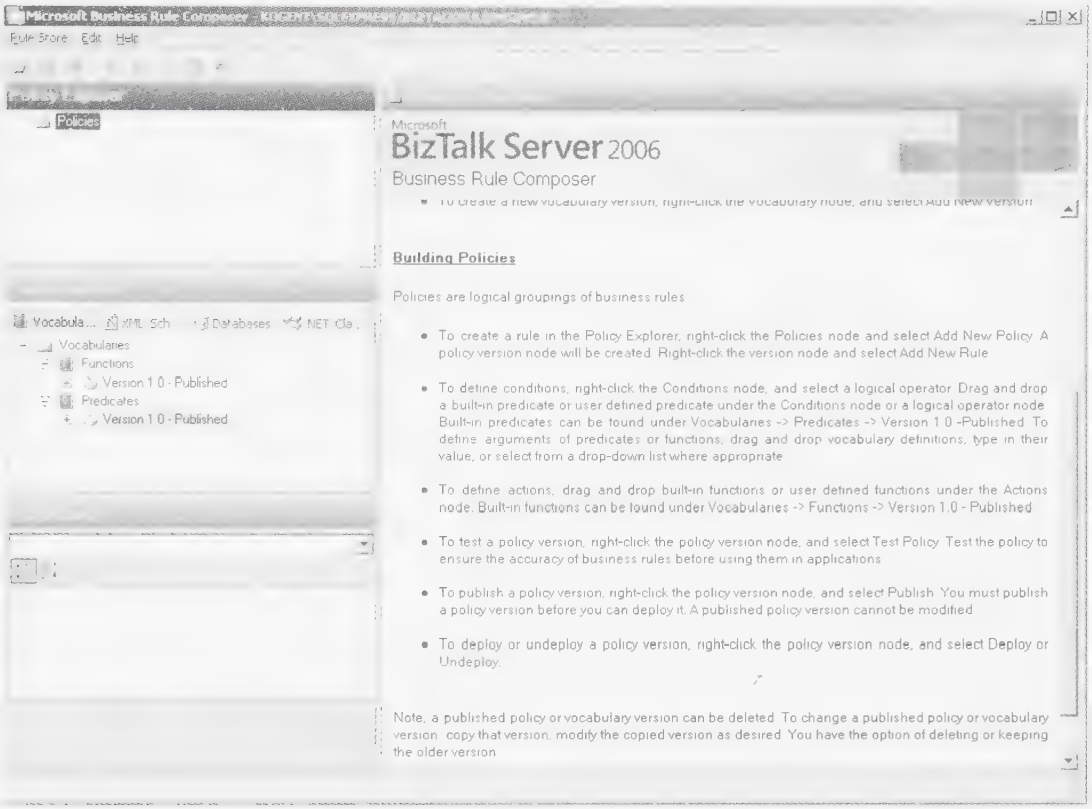


Fig.Biz-5.8

Sometimes, there is a need to load the policies from the **BizTalkRuleEngineDb** database, if it is not loaded by default. Otherwise, you can load the polices by using the **Rule Store→Load** menu options in Fig.Biz-5.8.

Let's now learn how to create a policy.

Creating a Policy

You can create policies for any organization by using the Business Rule Composer development tool provided by BZ 2006. BRC helps you to create and save versions of a policy simultaneously. For example, let's suppose you want to create a policy, Policy1, with the rule, PRule. You want the PRule rule to display the OrderedProductPrice as 5000, if the OrderedProductQuantity is more than 100. To create the Policy1 policy and the PRule rule, follow these steps:

1. *Click* **Start→All Programs→Microsoft BizTalk Server 2006→Business Rule Composer** to open the **Business Rule Composer** screen, as shown in Fig.Biz-5.9.

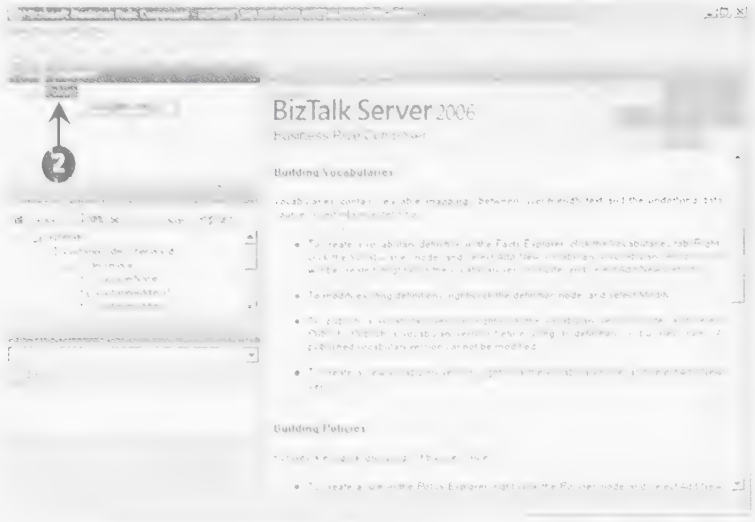


Fig.Biz-5.9

2. *Right-click* the **Policies** node in the **Policy Explorer** pane to display context menu (Fig.Biz-5.9).
3. Select the **Add New Policy** option to add a subnode **Policy1** in the **policies** node that you want to create, as shown in Fig.Biz-5.10.

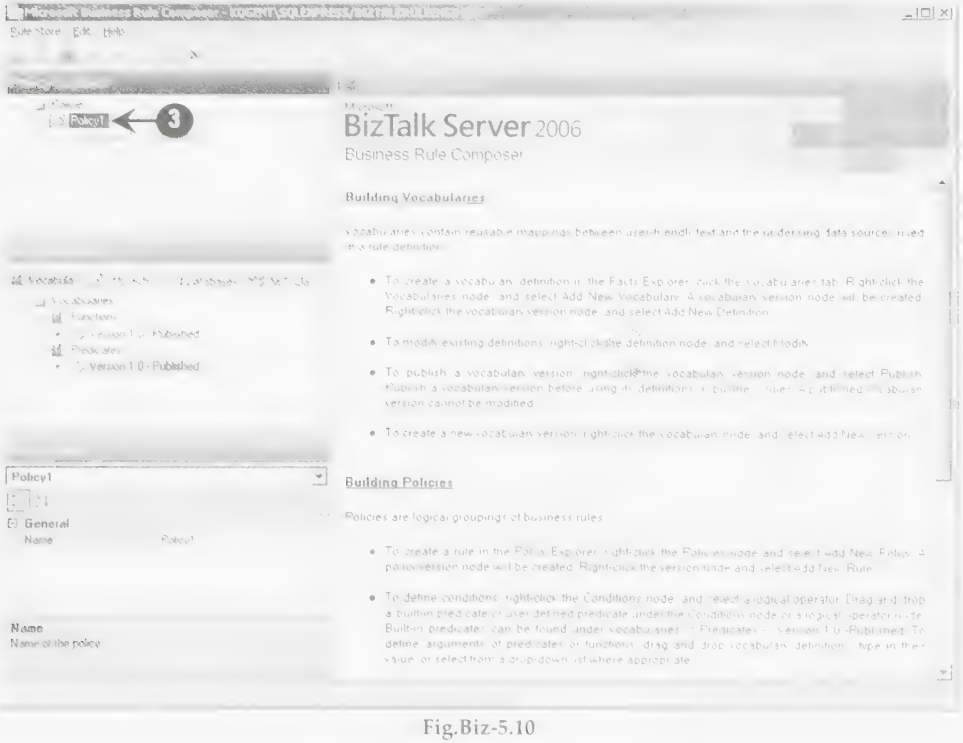


Fig.Biz-5.10

4. Now, *click* the **XML Schemas** tab in the **Facts Explorer** pane to display the **Schemas** node, as shown in Fig.Biz-5.11.

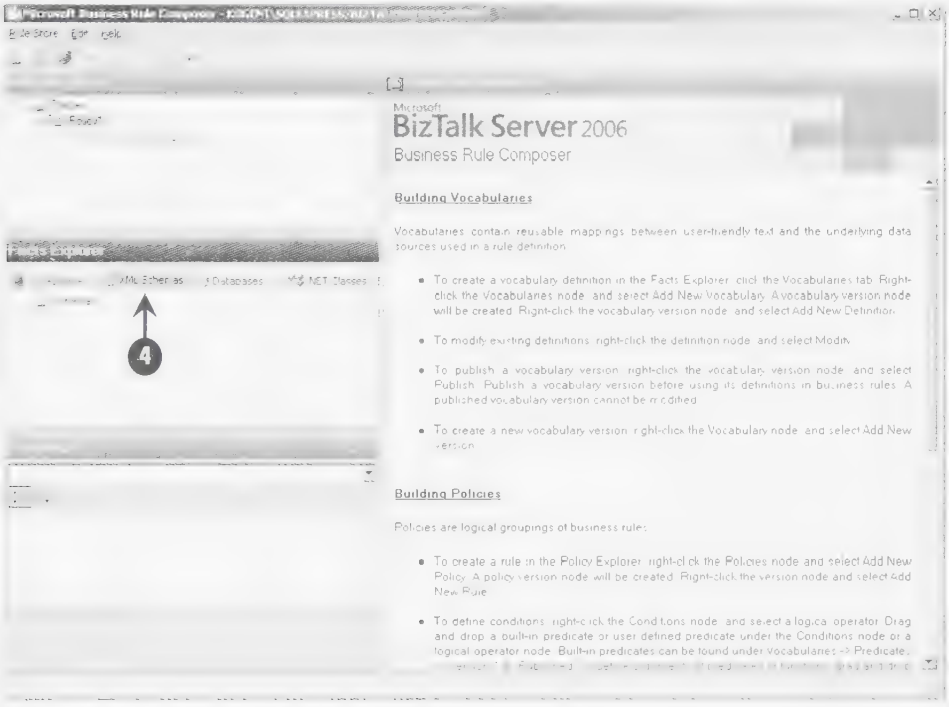


Fig.Biz-5.11

5. *Right-click* the **Schemas** node to display a context menu and *select* the **Browse** option to display the **Schema Files** dialog box, as shown in Fig.Biz-5.12.

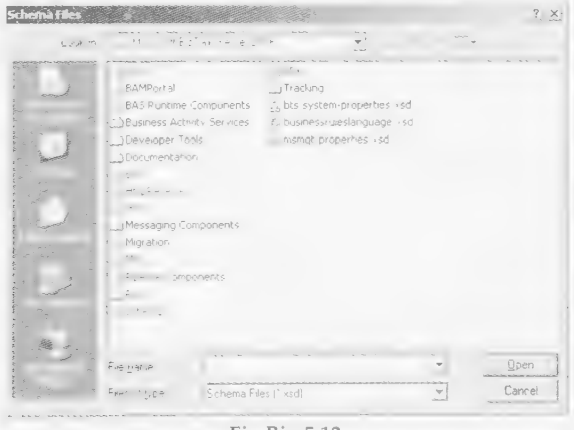


Fig.Biz-5.12

6. *Locate* and *select* the schema file that you want to attach to the Policy1 policy. In our case, we have used the **CustomerOrderSchema.xsd** file. The code for the **CustomerOrderSchema.xsd** xml file is shown in Listing 5.1

Listing 5.1: CustomerOrderSchema.xsd

```
<?xml version="1.0" encoding="utf-16"?>
<xs:schema xmlns:b="http://schemas.microsoft.com/BizTalk/2003"
xmlns="http://RulesTest.CustomerOrderSchema"
targetNamespace="http://RulesTest.CustomerOrderSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="OrderService">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="CustomerName" type="xs:string" />
        <xs:element name="CustomerAddress1" type="xs:string" />
        <xs:element name="CustomerAddress2" type="xs:string" />
        <xs:element name="CustomerCity" type="xs:string" />
        <xs:element name="CustomerState" type="xs:string" />
        <xs:element name="CustomerZip" type="xs:string" />
        <xs:element name="CustomerContactPhone" type="xs:string" />
        <xs:element name="OrderedProductNumber" type="xs:string" />
        <xs:element name="OrderedProductPrice" type="xs:int" />
        <xs:element name="OrderedProductQuantity" type="xs:int" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

7. *Click* the **Open** button (Fig.Biz-5.12) to attach the located schema to **Policy1**. The **CustomerOrderSchema.xsd** subnode appears under the **Schemas** node in the **Facts Explorer** pane as shown in Fig.Biz-5.13.
8. *Change* the **Document Type** property of **CustomerOrderSchema.xsd** to **RulesTest.CustomerOrderSchema**, as shown in the Fig.Biz-5.13.

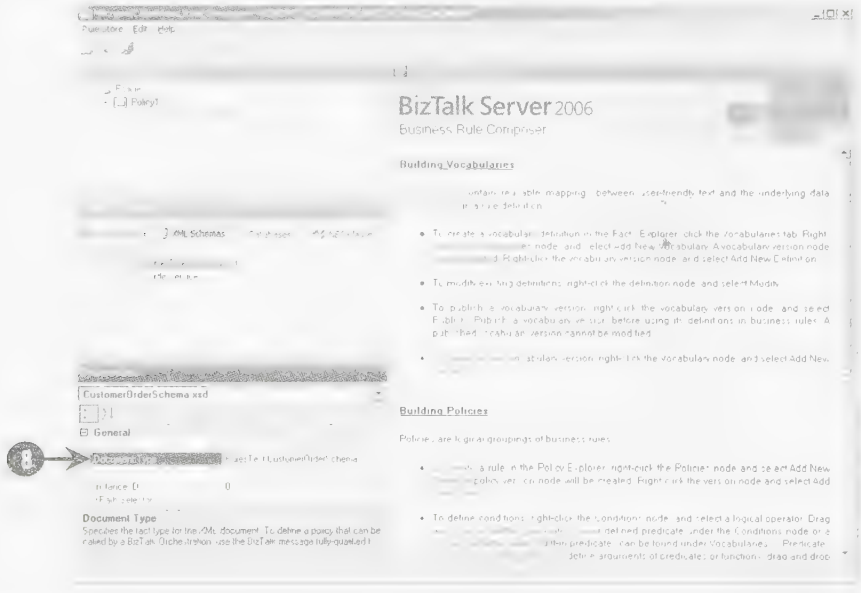


Fig.Biz-5.13

9. *Right-click* the **Version 1.0 (not saved)** subnode under the **Policy1** node in the **Policy Explorer** pane to display a context menu as shown in Fig.Biz-5.14.

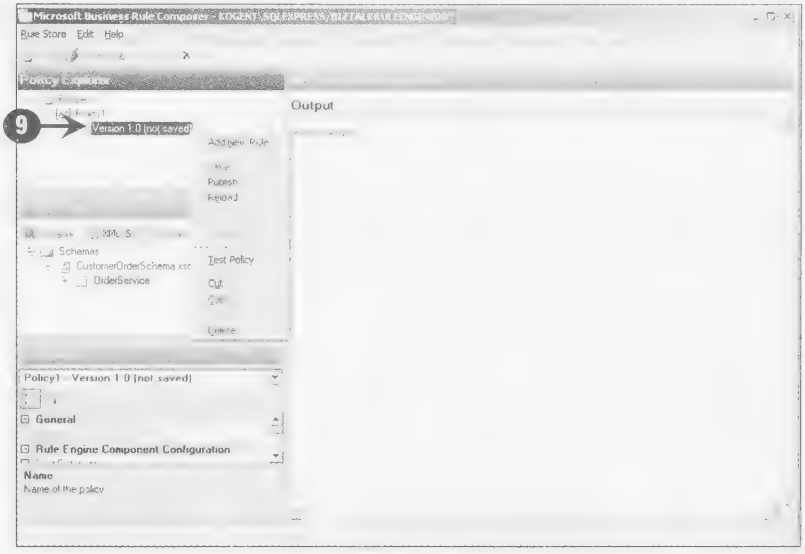


Fig.Biz-5.14

10. *Select* the **Add New Rule** option to add the **Rule1** subnode of the rule that you want to add to the **Policy1** policy (Fig.Biz-6.14). The **Rule1** subnode is added in the **Version 1.0 (not saved)** node to the right side of Policy Explorer pane. *Change* the name **Rule1** to **PRule**, as shown in Fig.Biz-5.15.

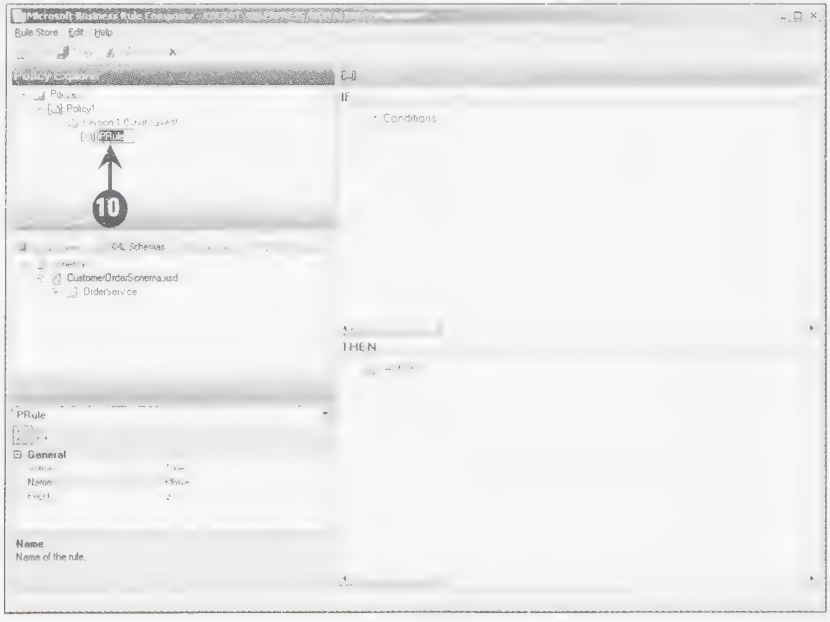


Fig.Biz-5.15

Fig.Biz-5.14 shows the **Policy1 - Version 1.0 - PRule** pane, which contains the IF and THEN panes. You can use the IF pane to specify a condition for a rule. The THEN pane allows you to specify the action to be performed, if the condition specified in the IF pane is satisfied.

Now, it time to provide the condition statement in the IF and Then panes. To provide the condition statement, perform the following steps:

1. *Right-click* the **Conditions** node in the IF pane to display a context menu as shown in Fig.Biz-5.16.
2. *Select* the **Predicates** option to display the **Predicates** menu (Fig.Biz-5.16).

Fig.Biz-5.16

Fig.Biz-5.15 shows the **Predicates** menu that contains various predicates, such as **Equal** and **GreaterThan**. You can use these predicates to specify various conditions for a rule.

3. *Select* the **GreaterThan** option from the **Predicates** menu. The arguments of the **GreaterThan** predicate appear in the **IF** pane (Fig.Biz-5.16).

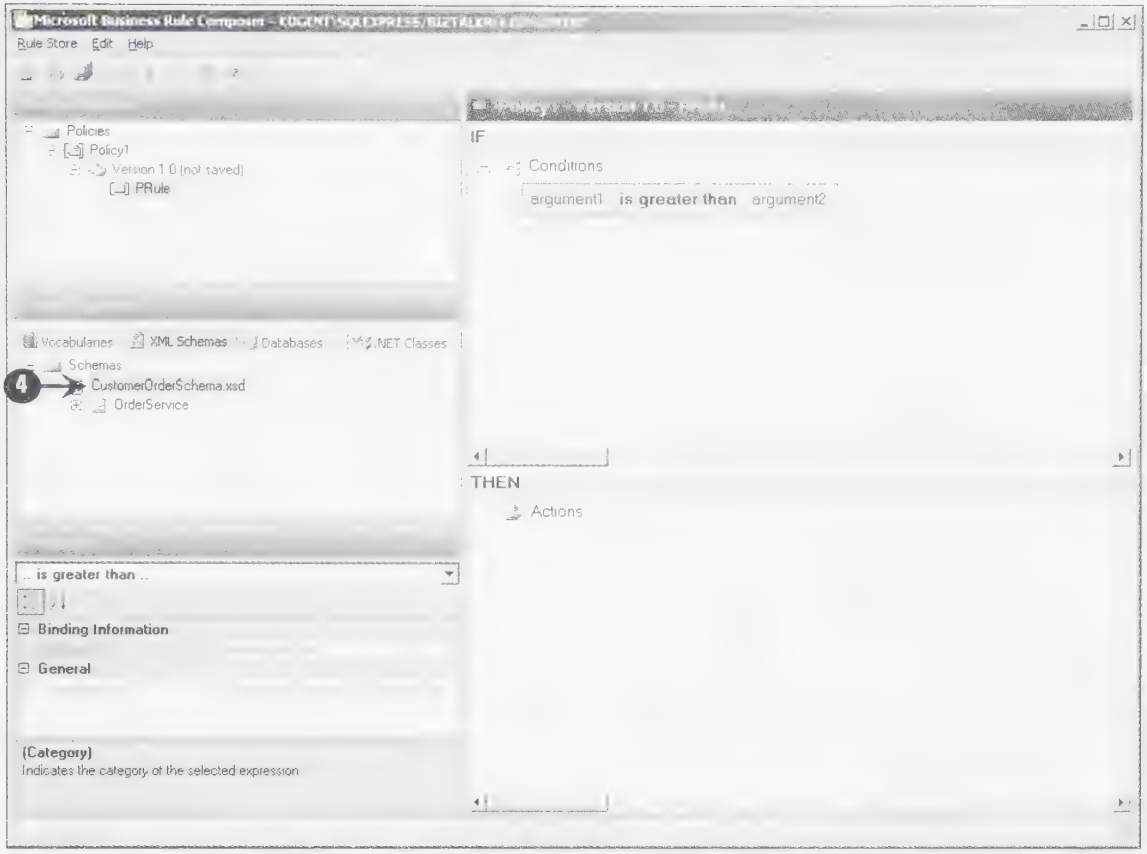


Fig.Biz-5.17

Fig.Biz-5.17 shows two arguments, **argument1** and **argument2**, of the **GreaterThan** predicate under the **Conditions** node in the **IF** pane.

4. *Expand* the **OrderService** subnode under the **CustomerOrderSchema.xsd** node in the **Fact Explorer** pane (Fig.Biz-5.17) to display various elements, such as **CustomerName** and **OrderedProductQuantity** of the **CustomerOrderSchema** schema, as shown in Fig.Biz-5.18.
5. *Drag* the **OrderedProductQuantity** element to the **argument1** argument in the **IF** pane (Fig.Biz-5.18).
6. *Click* the **arugument2** argument and enter the value (100) with which you want to compare the value of **OrderedProductQuantity** element (Fig.Biz-5.18).
7. *Drag* the **OrderedProductPrice** element from the **Facts Explorer** pane to the **THEN** pane. The **<enter a value>** argument appears in the **THEN** pane (Fig.Biz-5.18).

Fig.Biz-5.18

8. *Click* the **<enter a value>** argument and enter **5000**. The 5000 value is assigned to the **OrderedProductPrice** element. If the value of the **OrderedProductQuantity** element is greater than 100, as shown in Fig.Biz-5.19.

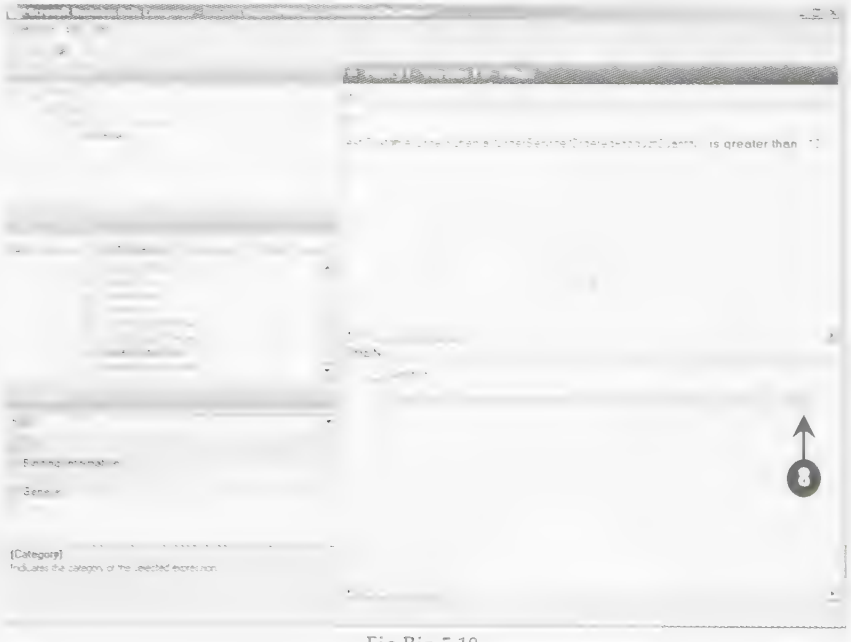


Fig.Biz-5.19

9. *Right-click* the **Version 1.0 (not saved)** submode in the **Policy Explorer** pane to display a context menu as shown in Fig.Biz-5.20.

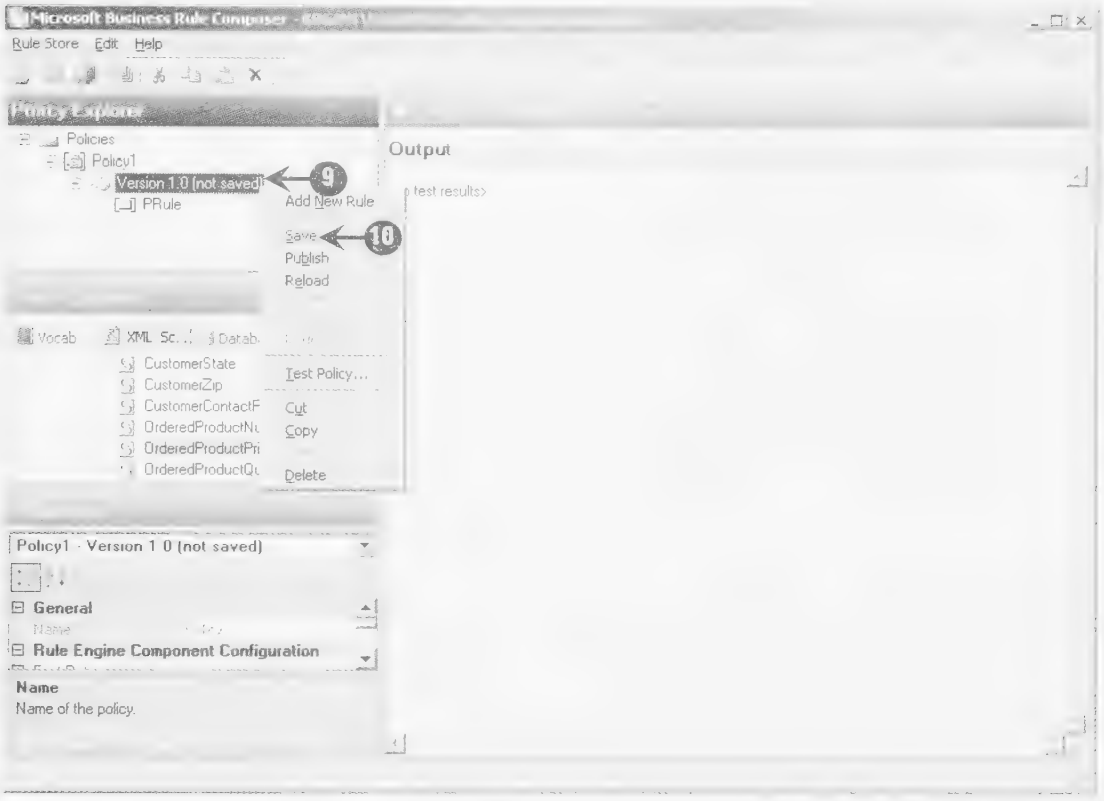


Fig.Biz-5.20

10. *Select* the **Save** option to save the current version of the **Policy1** policy in the BizTalk Server database (Fig.Biz-5.20).
11. *Close* the **Business Rule Composer** screen.

Now after creating the **PRule** rule under **Policy1**, it's time to publish and test the **Policy1** policy.

Publishing and Testing a Policy

After creating the PRule rule under Policy1, you need to publish the Policy1 and test it. Publish the Policy1 enables you to not change the version of policy, this helps to manage the version of policy. Testing enables you to verify whether the rules in the policy are implemented properly. To publish and test the Policy1 policy, follow these steps:

1. *Open* the **Business Rule Composer** screen.
2. *Right-click* the **Version 1.0** submode under the **Policy1** node to display a context menu as shown in Fig.Biz-5.21

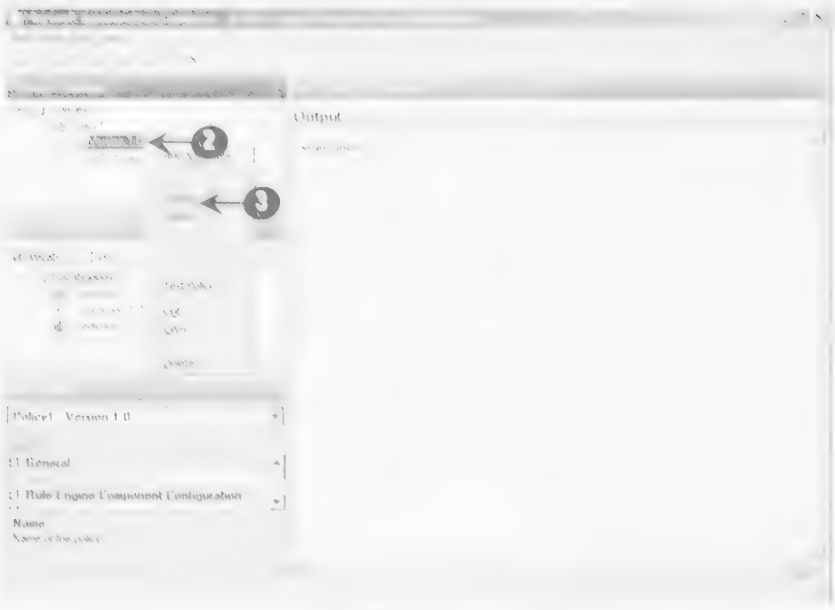


Fig.Biz-5.21

3. *Select* the **Publish** option to publish **Policy1**. On doing this, the name of **Version 1.0** subnode changes to **Version 1.0 - Published**, as shown in Fig.Biz-5.22

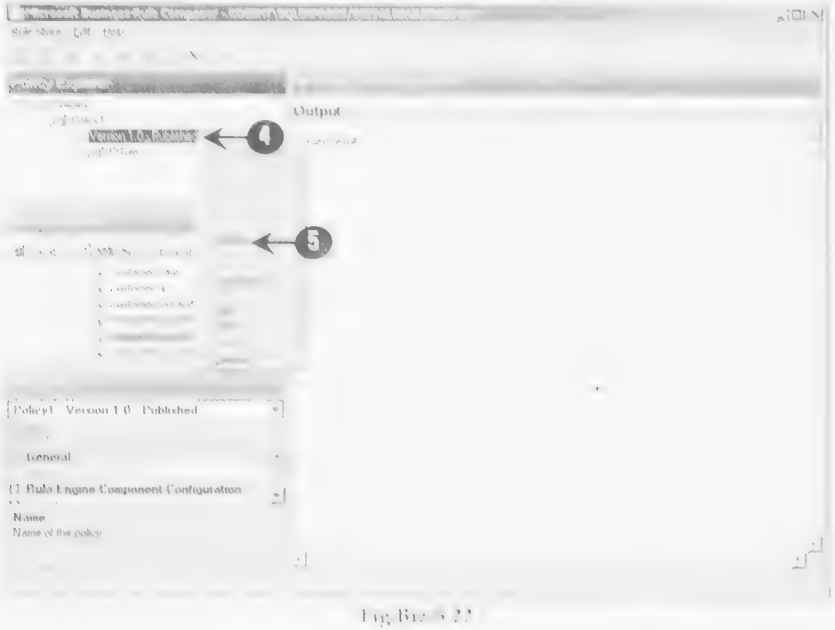


Fig.Biz-5.22

4. *Right-click* the **Version 1.0 - Published** node to display a context menu as shown in Fig.Biz-5.23

5. *Select* the **Test Policy** option from the context menu (Fig.Biz-5.22) to display the **Select Facts** dialog box, as shown in Fig.Biz-5.23.

Fig.Biz-5.23

This Fig.Biz-5.23 shows the **Select Facts** dialog box, which contains the **Add instance** button that you can use to attach an XML file to the **Sample Rule** policy. The **Add instance** button is deactivated by default, when you select a valid XML file this button will be activated. The XML file enables you to test the **Policy1** policy.

6. *Select* the **RulesTest.CustomerOrderSchema** subnode under the **XML Documents** node to attach the XML file to the **CustomerOrderSchema** schema (Fig.Biz-5.23). The **Add Instance** button is enabled after you select the **RulesTest.CustomerOrderSchema** node (Fig.Biz-5.25).
7. *Click* the **Add instance** button to display the **Open** dialog box as shown in Fig.Biz-5.24.

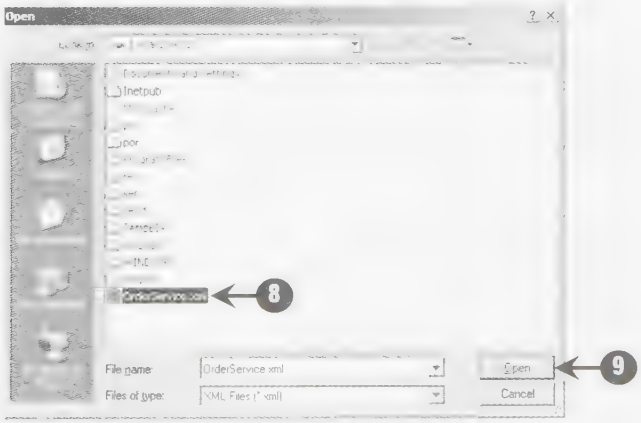


Fig.Biz-5.24

8. *Locate* and *select* the **XML** file that you want to use to test the Policy1 policy (Fig.Biz-5.24).

9. *Click* the **Open** button in the **Open** dialog box (Fig.Biz-5.24). The location and name of the XML file appears under the **CustomerOrderSchema** node in the **Select Facts** dialog box, as shown in the Fig.Biz-5.25

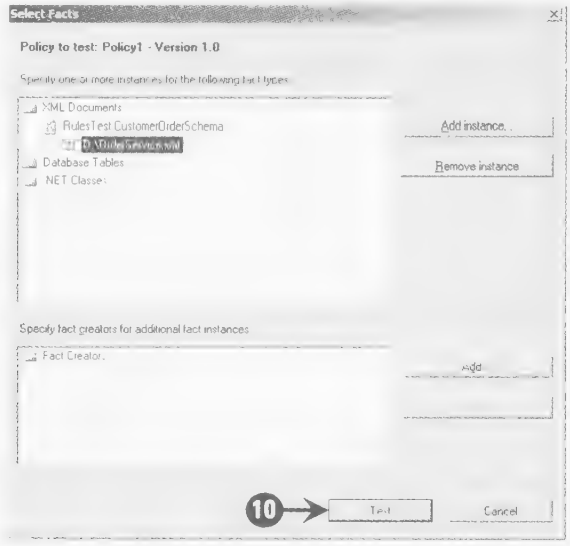


Fig.Biz-5.25

10. *Click* the **Test** button (Fig.Biz-5.25) to display **Test Result** to **True** in the **Output** pane of the **Business Rule Composer** screen, as shown Fig.Biz-5.26.

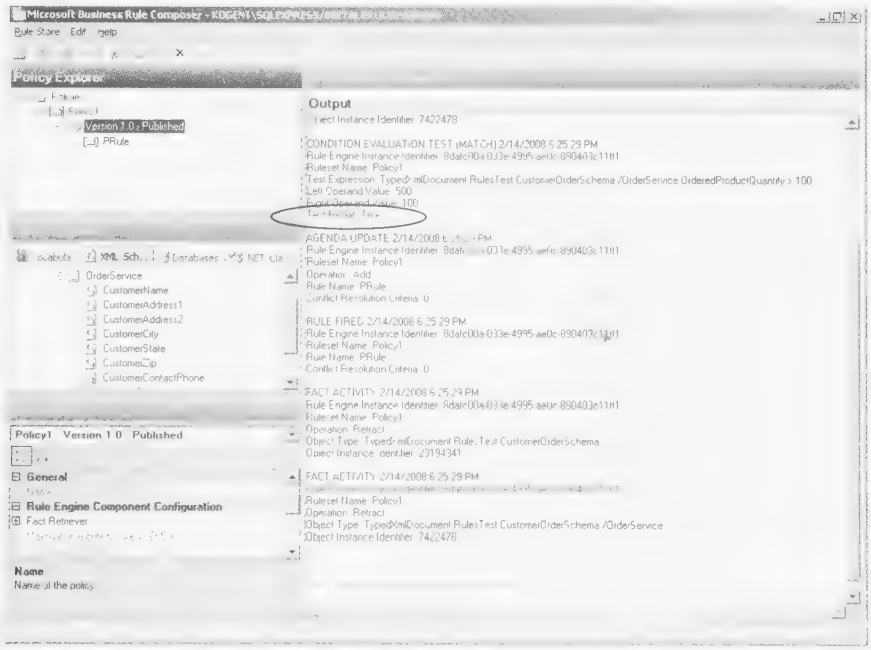


Fig.Biz-5.26

In Fig.Biz-5.26, you can see the **Output** pane, which shows that the test result is True, in the encircled area. Now, you can verify the XML file for the changed **OrderedProductPrice** value. The updated value of **OrderedProductPrice** in the **OrderService.xml** is shown in Listing 5.2.

Listing 5.2: Updated OrderService.xml file

```
<?xml version="1.0" encoding="utf-16"?>
<ns0:OrderService xmlns:ns0="http://RulesTest.CustomerOrderSchema">
  <CustomerName>CustomerName_0</CustomerName>
  <CustomerAddress1>CustomerAddress1_0</CustomerAddress1>
  <CustomerAddress2>CustomerAddress2_0</CustomerAddress2>
  <CustomerCity>CustomerCity_0</CustomerCity>
  <CustomerState>CustomerState_0</CustomerState>
  <CustomerZip>CustomerZip_0</CustomerZip>
  <CustomerContactPhone>CustomerContactPhone_0</CustomerContactPhone>
  <OrderedProductNumber>OrderedProductNumber_0</OrderedProductNumber>
  <OrderedProductQuantity>500</OrderedProductQuantity>
  <OrderedProductPrice>5000</OrderedProductPrice>
</ns0:OrderService>
***
```

Now, it's time to invoke the policy in the BizTalk application.

Calling the Policy from an Orchestration

After creating, publishing and testing the policy, it's time to apply the policy to the BizTalk project. This is required so that the messages get exchanged on the basis of the rule mentioned in the policy. You can call the **Policy1** policy from an orchestration. For creating an orchestration, you need to develop a BZ project as discussed earlier in Chapter 3. A BZ project can be developed by using the following steps:

1. *Click* **Start→Programs→Microsoft Visual Studio 2005→Microsoft Visual Studio 2005** to open the **Start Page** of the Microsoft Visual Studio window.
2. *Select* **File→New→Project** on the **Start Page** of the Microsoft Visual Studio window to open the **New Project** dialog box.
3. *Select* **BizTalk Projects** from the **Project types** pane, and *select* the **Empty BizTalk Server Project** from the **Templates** pane.
4. Next, *enter* the name of the BizTalk project as **RulesTest** in the **Name** textbox.
5. Now, *click* the **OK** button to accept the settings. This opens the **RulesTest - Microsoft Visual Studio** window.
6. *Add* the **CustomerOrderSchema.xsd** schema file to the BizTalk project. To know how to add the **CustomerOrderSchema.xsd** schema file, please refer to section 'Adding Schema to the Project' of Chapter 4.
7. *Right-click* the **RulesTest** project in the **Solution Explorer** and *select* **Add→New Item** from the context menu to open the **Add New Item – RulesTest** dialog box.
8. In the **Add New Item – RulesTest** dialog box, *select* the **Orchestration Files** node from the **Categories** pane and select **BizTalk Orchestration** from the **Templates** pane.
9. *Enter* the name of the orchestration as **Customer.odx** in the **Name** text box.

10. Now, *click* the **Add** button to add the **Customer** orchestration item to the **RulesTest** project. This opens the **RulesTest – Microsoft Visual Studio** screen.
11. *Set* the **Transaction type** property of the **Customer** orchestration to **Atomic**. This is required for calling the policy and enabling the **Call Rules** shape in the Toolbox.
13. Now, *right-click* the **Messages** folder on the **Orchestration View** pane to open the context menu.
14. *Select* the **New Message** option from the context menu to add a message to the **RulesTest** project. The **Message_1** message is added to the **Messages** folder.
15. *Change* the properties of the **Message_1** message from the **Properties** pane according to values mentioned in Table 5.1.

Table 5.1: Properties and values of Message_1 message

Property	Value
Identifier	Corder
Message Type	RulesTest.CustomerOrderSchema (Schema Message)

After applying these message properties, you need to add the port type by using the following steps:

1. In the **Orchestration View** pane, expand the **Types** node.
2. *Right-click* the **Port Types** folder and *select* the **New One-way Port Type** option from the context menu to add a new port type named **PortType_1**.
3. Now, expand the **PortType 1** subnode to show the **Operation_1** subnode. Expand the **Operation_1** subnode and *select* the **Request** subnode.
4. *Change* the **Message Type** property to **RulesTest.CustomerOrderSchema** (Schema node) from the **Properties** pane.

After adding port type, you need to add the ports by using the following steps:

1. In the **Orchestration View** pane, *right-click* the **Ports** folder and *select* the **New Port** option from the context menu to add a new port named **Port_1**.
2. Now, *right-click* the **Ports** folder in the **Orchestration View** pane to add another port.

In this project, we need two ports. One port will be used to send messages from the Send shape and the other port will be used to receive messages from the Receive shape. The Send/Receive shapes will be discussed later in this chapter.

3. After this, *change* the properties of **Port_1** and **Port_2** ports in the **Properties** pane according to the values mentioned in the Table 5.2 and Table 5.3.

Table 5.2: Properties for Port_1 port

Property	Value
Port Type	RulesTest.PortType_1
Communication Direction	Receive

When you set the value of port type is **RulesTest.PortType_1** as mentioned in Table 5.2, it is necessary to select the **Port_1** port and set its properties as well.

Table 5.3: Properties for Port_2 port

Property	Value
Port Type	RulesTest.PortType_1
Communication Direction	Send

When you set the value as **RulesTest.PortType_1** of port type property as mention in Table 5.3, it is necessary to select the **Port_2** port and set its properties in the Port Surface area.

Now, the **Port_1** port and **Port_2** port is added in the **Ports** node.

We need to add the Receive and Send shapes for exchanging messages using the ports defined. Shapes can be added using the following steps:

1. *Drag* the Receive and Send shapes (from the BizTalk Orchestrations components on the Toolbox) onto the Orchestration Surface area where the point labeled **Drop a shape from the toolbox here** is indicated.
2. *Change* the properties of the **Receive_1** and **Send_1** shapes in the Properties pane according to values in Table 5.4 and Table 5.5.

Table 5.4: Property Values for Receive_1 shape

Property	Value
Activate	True
Message	Corder
Operation	Port_1.Operation_1.Request

Table 5.5: Property Values for Send_1 shape

Property	Value
Message	Corder
Operation	Port_2.Operation_1.Request

After entering these values, you will be presented with screen, as shown in Fig.Biz-5.27.

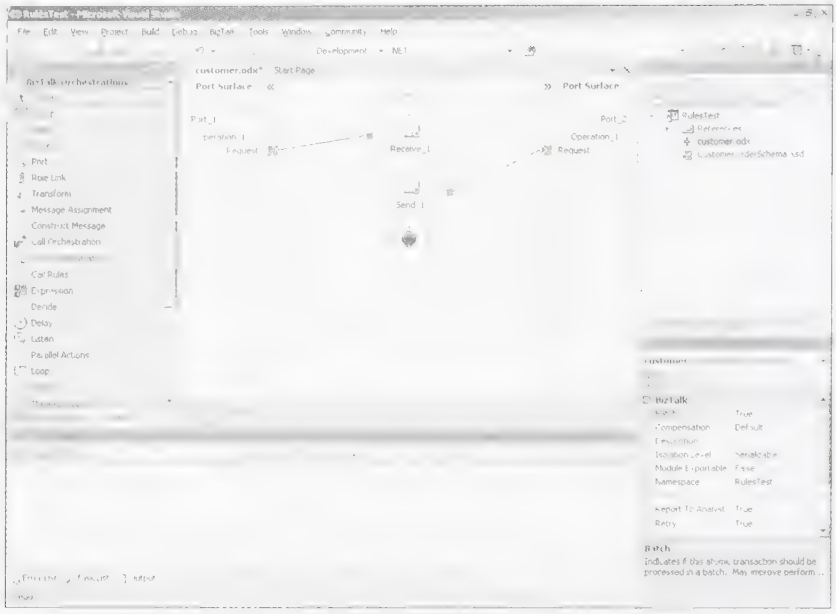


Fig.Biz-5.27

Now, we need to add the Call Rules shape, which is required for calling the **Policy1** policy defined earlier. You can add the **Call Rules** shape by dragging the same from the Toolbox and drop it between **Receive_1** and **Send_1** shape on the **Customer** orchestration, as shown in the Fig.Biz-5.28

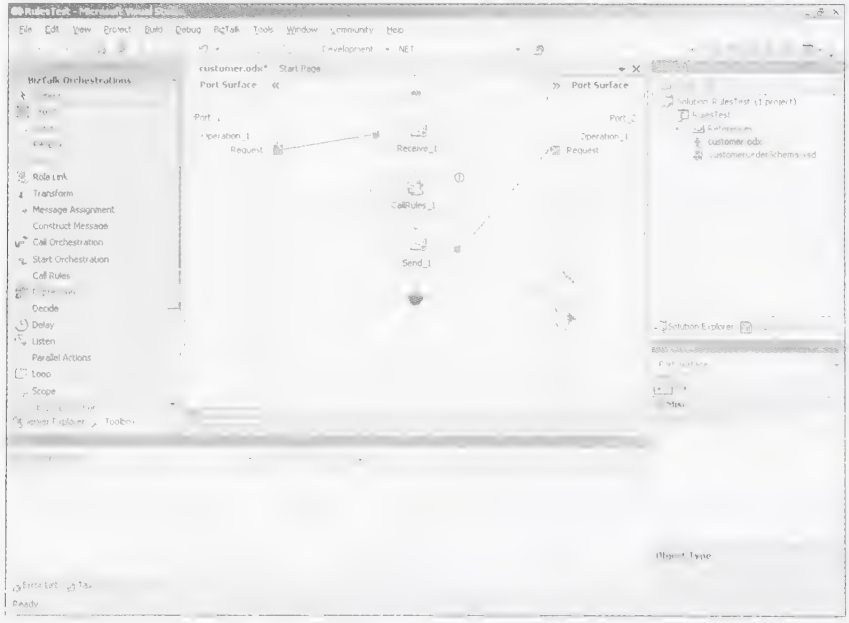


Fig.Biz-5.28

3. Now, you need to configure the **Call Rules** shape. To configure this shape, you *double-click* on the **CallRules_1** shape to open the **CallRules policy configuration** dialog box, as shown in the Fig.Biz-5.29.

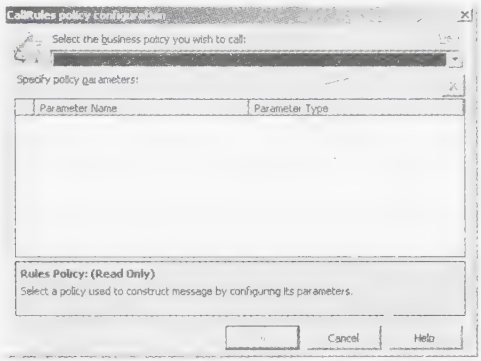


Fig.Biz-5.29

4. *Select* **Policy1** from the **Select the business policy you wish to call** drop-down menu, as shown in Fig.Biz-5.30.

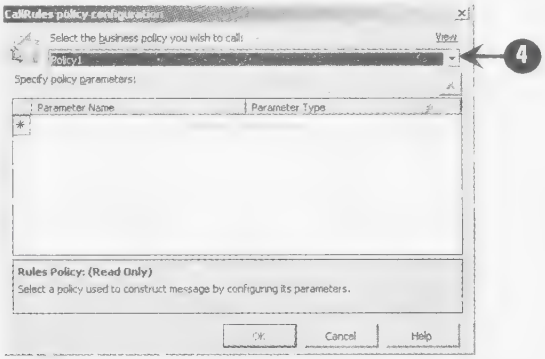


Fig.Biz-5.30

5. Now, *click* **click here to add a new row!** tab (Fig.Biz-5.30) and set **Parameter Name** as **Corder**, as shown in the Fig.Biz-5.31.

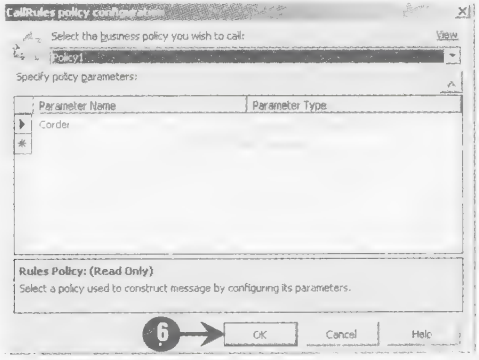


Fig.Biz-5.31

6. Click the **OK** button (Fig.Biz-5.31) to accept the settings. This will also open the **RulesTest-Microsoft Visual Studio** screen, as shown in Fig.Biz-5.32.

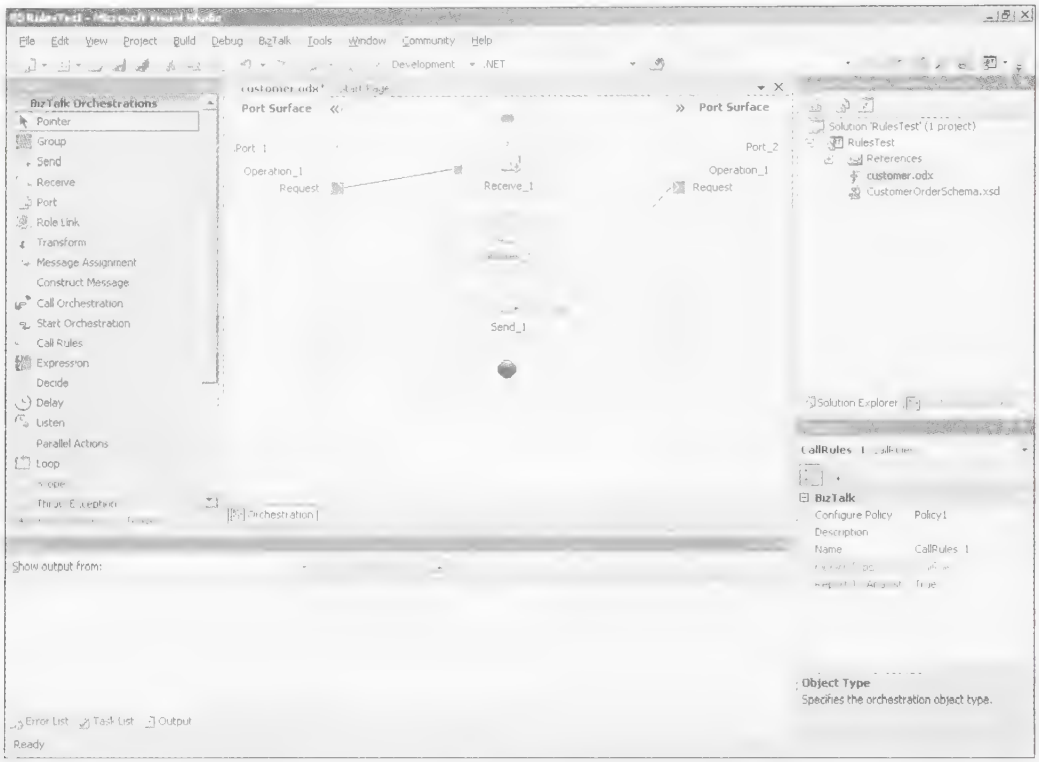


Fig.Biz-5.32

Now, assign a strong name to the assembly and then deploy the project as discussed under section 'Assignment of Strong Name to the Assembly' and 'Deploying the Project' in Chapter 3. After deploying the project, you need to test the project. But before testing the project, you need to configure the project. You can configure project through the **BizTalk Server 2006 Administration Console**. For configuring the project, you need to create the physical Send/Receive ports and locations, which are required for configuring the application in BizTalk Server using the BizTalk Server 2006 Administration Console. After this, bind the physical and logical ports and start the deployed orchestration. You also need to deploy the policy created using the BRC so that rule is applied to messages, which are getting transferred using the **RulesTest** application.

Note

Ensure that you are not using the passThruReceive pipeline for the receive location. You need to use the default XMLReceive and XMLTransmit pipelines for this application.

Deploying the Policy and Testing the Application

Deploying a policy makes your policy available to your project. After deploying a policy, you need to test this policy by using an XML file in your BZ application. The created policy can be deployed by the following steps:

1. *Click* **Start→All Programs→Microsoft BizTalk Server 2006→Business Rule Composer** to open the **Business Rule Composer** window, as shown in Fig.Biz-5.33.

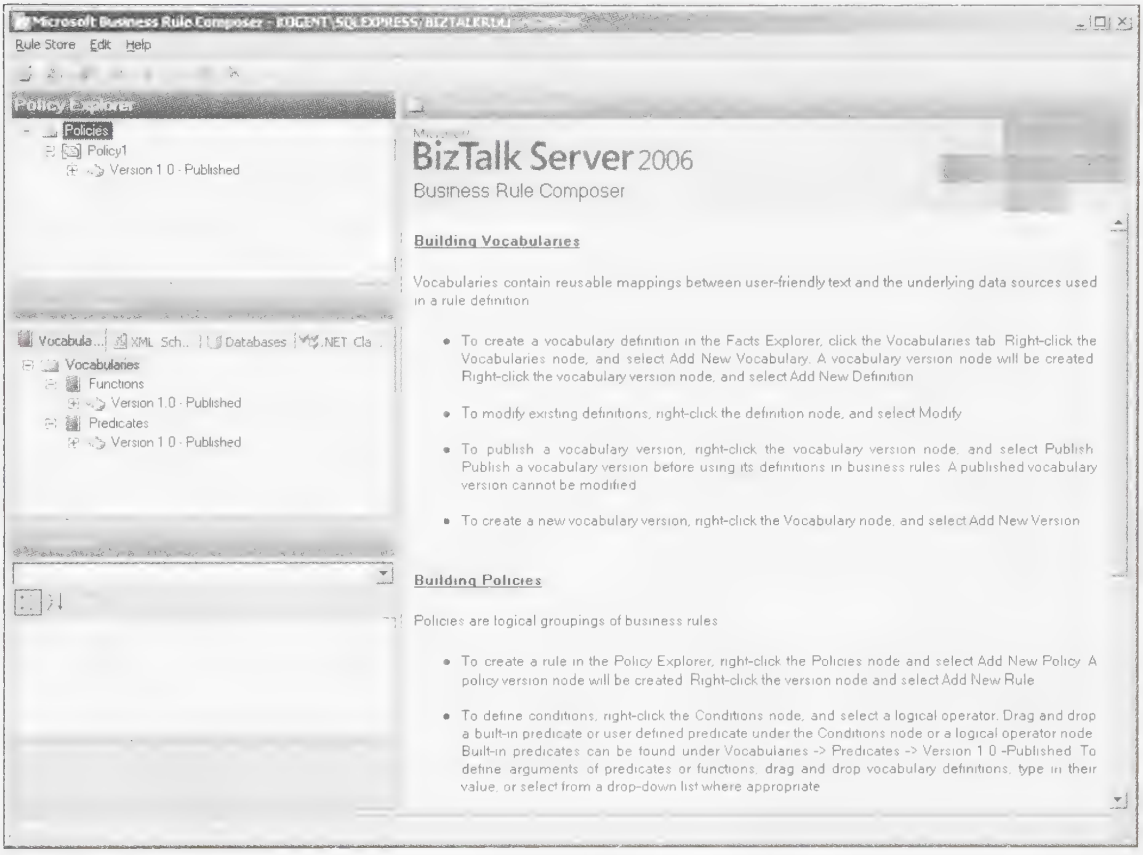


Fig.Biz-5.33

2. *Right-click* the **Version 1.0-Published** node to open the context menu and select the **Deploy** menu option to deploy **policy1** (Fig.Biz-5.22). This will open the screen, as shown in Fig.Biz-5.34.

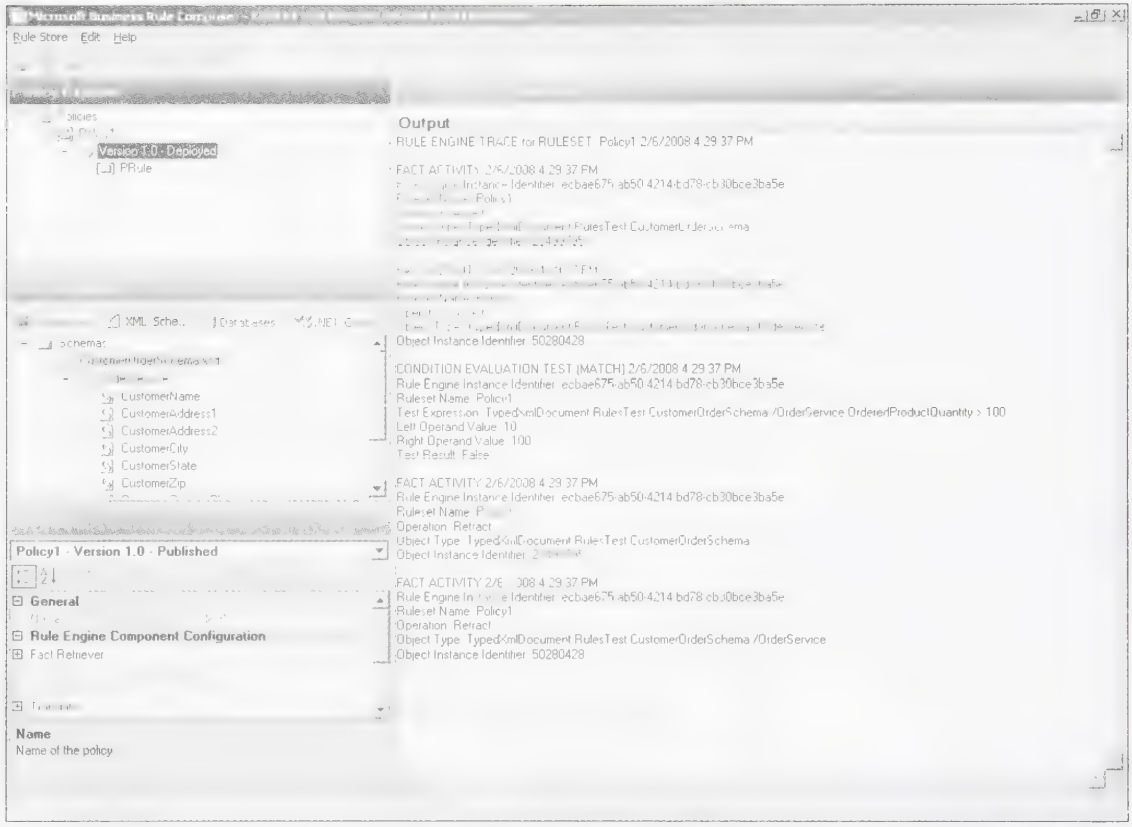


Fig.Biz-5.34

In Fig.Biz-5.34, you can see that the **Version 1.0-Published** node has changed to **Version 1.0-Deployed**.

3. *Close* the BRC.

Now it's time to test the **RulesTest** project. The project can be test by the following steps:

1. Create an XML file named as **OrderService.xml** with the code as shown in Listing-5.3.

Listing 5.3: OrderService.xml file

```
<?xml version="1.0" encoding="utf-16"?>
<ns0:OrderService xmlns:ns0="http://RulesTest.CustomerOrderSchema">
  <CustomerName>CustomerName_0</CustomerName>
  <CustomerAddress1>CustomerAddress1_0</CustomerAddress1>
  <CustomerAddress2>CustomerAddress2_0</CustomerAddress2>
  <CustomerCity>CustomerCity_0</CustomerCity>
  <CustomerState>CustomerState_0</CustomerState>
  <CustomerZip>CustomerZip_0</CustomerZip>
  <CustomerContactPhone>CustomerContactPhone_0</CustomerContactPhone>
  <OrderedProductNumber>OrderedProductNumber_0</OrderedProductNumber>
  <OrderedProductQuantity>10</OrderedProductQuantity>
  <OrderedProductPrice>0</OrderedProductPrice>
</ns0:OrderService>
```

2. Now, place the **OrderService.xml** file in the received location mentioned during the creation of a physical receive port. In our case, it is "D:\receiveORDER".
3. It will take sometime to move your XML file to the send location "D:\SendORDER" (mentioned during the creation of the physical send port). You can see the XML file will not have the changed value of the column **OrderedProductPrice** because **OrderedProductQuantity** is less than 100.
4. Now *change* the value of **OrderedProductQuantity** to 500 and save the **OrderService.xml** file and place the same in the "D:\receiveORDER" location.
5. When the file has moved to the send location "D:\SendORDER", open the XML file and you can see that that the value of **OrderedProductPrice** has changed to 5000 (as per the PRule rule defined in BRC).

With this, we come to the end of this chapter. What follows is a brief summary of the main points covered in the chapter.

Summary

In this chapter, we discussed BZ business rules and policies. We also discussed the installation of the Business Rule Composer and learned how to create, publish, deploy and test a policy by using an XML schema in detail. At the end of the chapter, we learned to call the policy from an orchestration by developing a BZ project.

In the next chapter, we will discuss B2B business processes and learn how to conduct online trading through the B2B e-commerce process.

Chapter 6

Overview of B2B Process

In this Chapter

- The B2B Process
- Open Buying on the Internet (OBI)

A Business-to-Business (B2B) process is a term used to describe electronic commerce transactions between businesses. It is used for exchanging products or services among different companies. Companies conduct B2B transactions in an online market, which involves buy-side and sell-side solutions in addition to trading exchanges of online products and marketplaces. Buy-side and sell-side solutions are a set of applications that allow companies to purchase or sell their products or services online. Business protocols are sets of rules in data exchange format that are implemented using different standards, such as EDI, Open Catalog Interface (OCI) and Commerce eXtensible Markup Language (cXML). These business protocols are used by companies to communicate with each other and perform B2B transactions. Business protocols specify the format for messages, such as purchase orders (POs) and invoice in each standard used for performing the B2B process.

This chapter explains the traditional business process of conducting B2B transactions and B2B e-commerce. It also explains B2B exchange, B2B hub architecture, and buy-side and sell-side solutions. In addition, the chapter explains the data exchange formats that are used for communication in a B2B transaction.

The B2B Process

Nowadays, a majority of companies perform their business activities through B2B e-commerce instead of the traditional B2B system. In the traditional B2B process, companies face many problems such as high cost of business transactions, ineffective use of time to perform business activities, and lack of sufficient manpower. For example, in traditional B2B processes, companies use traditional methods, such as telephone and fax to perform their business activities. These traditional methods require more time and are costlier to perform as compared to companies that use B2B e-commerce methods, such as the Internet to conduct their business activities.

In B2B e-commerce, companies perform their business activities with trading partners and supplier companies by using the Internet or dedicated Virtual Private Network (VPN) links. Companies automate business activities by using different packaged and web applications that provide online transaction of products and services within quick response time and are cost effective too. Automating business activities help companies maintain a consistent and effective relationship with their trading partners.

Now, we will discuss traditional B2B process in detail.

The Traditional B2B Process

In the traditional B2B process, different employees of a company perform their business, such as preparing of PO, sending of invoice and preparation of bill. For example, an employee of a supplier company sends Request For Quotations (RFQs) that are compiled by an employee in the buyer company. Another employee in the buyer company prepares the PO and sends it to the supplier company. Fig.Biz-6.1 shows the traditional B2B process:

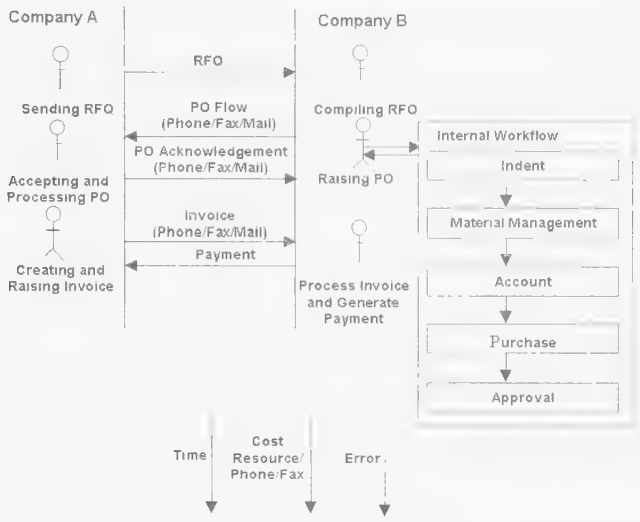


Fig.Biz-6.1

This figure shows the traditional B2B process in which different employees of Company A and Company B perform various tasks, such as sending POs, receiving invoices from the partner, constructing a payment voucher of the partner, and receiving the payment. The traditional business process has following limitations:

- Communication between trading partners and supplier companies are performed through traditional systems, such as telephone, mail, and fax.
- Increased overhead costs of business activities, because a company needs large number of employees to perform its business activities.
- Time-consuming process, because all the tasks are performed manually by different departments of the company.
- All the tasks are performed manually by the employees of the company. Thus, errors can occur in tasks such as preparing PO, sending invoices, and preparing bills.

After understanding traditional B2B process, let's understand the concept of the B2B e-commerce process.

The B2B E-Commerce Process

In the B2B e-commerce process, a set of products and services facilitates the exchange of products, services and information over electronic networks within a company, and between companies and their customers. To conduct B2B e-commerce, companies integrate with buyers and suppliers and perform business activities such as buying and selling of products or services by using electronic systems, such as software applications and computers. To perform B2B e-commerce:

1. The buyer company first connects to a B2B exchange using the Internet. A B2B exchange is an online market place where companies buy and sell products and services from and to other companies.

2. The buyer company looks at product catalogs of various supplier companies for products or services that it wants to buy.
3. After looking at the catalogs, the buyer company identifies the supplier companies that offer good quality products at competitive prices. Having done this, the buyer company sends POs to the concerned supplier companies.
4. The supplier companies send an acknowledgment for the receipts of the POs and prepare invoices for them. The invoices and advance shipment notifications are sent to the buyer company.
5. The buyer company receives the invoices and sends an acknowledgement for them to the supplier companies. It then processes the invoices and sends payment to the supplier companies.

Fig.Biz-6.2 shows the B2B e-commerce process:

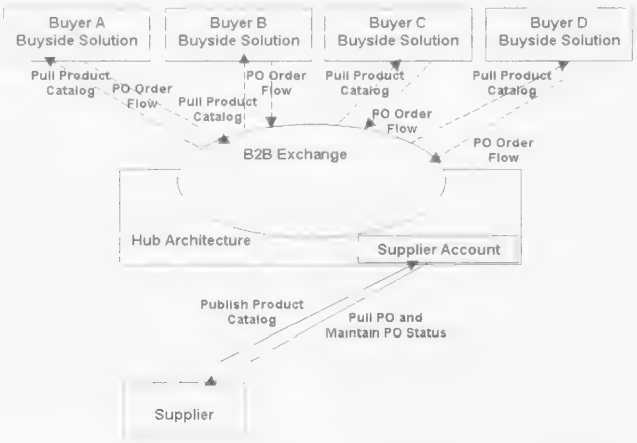


Fig.Biz-6.2

This figure shows the B2B e-commerce process in which buyer companies, such as Buyer A and Buyer B, connect to the B2B exchange to access product catalogs of supplier companies. The various components involved in the B2B e-commerce process are:

- B2B exchange
- E-marketplace
- B2B partners
- Data exchange formats and specifications

Let's now look at each in detail.

B2B Exchange

B2B exchanges are simply online marketplaces where buyers and sellers offer products and services and conduct business transactions. It enable companies to easily find the goods they need, complete the transaction, and save money in the process by using the online shopping system through the Internet. Various types of B2B exchanges exist. Some of the common ones are:

- **Consortia**: This type of B2B exchange is used by a group of vendors of a specific industry. For example, Global Food Exchange is a consortia exchange where a group of vendors related with the food products industry perform business activities.
- **Public**: This type of B2B exchange is open to all companies. The public exchange can be used by any company that wants to sell and purchase products at competitive prices. A Customer can view their products on the websites of different companies and select those, which best suit his or her budget. Commerce One is an example of a public exchange.
- **Private**: This type of B2B exchange is run by a single company. The exchanges run by Wal-Mart and Dell are examples of a private exchange.
- **Vertical**: This type of B2B exchange is used by a specific industry. In this type of B2B exchange, services are provided to a single industry. Here, you can take the example of an exchange used by the steel industry, which includes all different levels within that industry, right from suppliers of raw material through the production of components and finished goods to customers.
- **Horizontal**: Horizontal type of B2B exchanges are used by multi-industries. These are useful when several industries have a common need for particular kinds of inputs, services or share particular problems. For example, an exchange related to stock market is a horizontal type of exchange.

E-Marketplace

E-marketplace is an online exchange that provides services, such as auctioning and tendering to buyer and supplier companies. It is a trading platform created with a collection of searchable web services to enable trading partners conduct business activities over the Internet. E-marketplaces provide a place for buyer and supplier companies to market their products and services. Web services provide a standardized way of integrating web-based applications using XML standards over the Internet. The various e-marketplace services are:

- Supplier directories and search engines
- Tendering
- Classifieds
- Auctions

Let's discuss these services one by one.

- **Supplier directories and search engines:** In supplier directories and search engine services, databases containing data related to supplier companies are maintained. A buyer company can use this service to identify supplier companies and buy products and services from them. A buyer company can use the databases to search for supplier companies selling the products they need.
- **Tendering**: The tendering services maintain a database containing data related to tenders declared by different buyer companies. There are two types of tendering services-- public and open. The public tendering service is used by government institutions to declare tenders for purchasing products and services. The open tendering service is used by a company to declare tenders for purchasing products and services.

- **Classifieds**: The classifieds service help a supplier company provide information related to its products that are for sale. A supplier company provides advertisements about its products in the magazines that are published on different websites in the form of classifieds.
- **Auctions**: The auction service helps a supplier company to sell its products or services by auctioning them to the highest bidding company. This helps a supplier company to obtain the highest prices for its products or services. The auction service helps a buyer company to find bargains for products or services it wants to purchase. The reverse auction service, on the other hand, enables a buyer company to find a supplier company that offers the lowest price for the products or services it wants to buy. In the reverse auction service, supplier companies continue to decrease the prices of their products and services to obtain contracts from buyer companies. Supplier companies offering the lowest prices obtain the contracts from the buyer company. The reverse auction service is also called sourcing or buyer auction.

B2B Partners

A B2B partner represents a company from which another company purchases or sells its products or services. A company and its B2B partners use buy-side and sell-side solutions to purchase or sell products and services in a B2B exchange and e-marketplace. Companies use the B2B hub architecture to exchange information with B2B partners. The architecture of a B2B hub is described next.

The B2B Hub Architecture

The B2B hub architecture is based on the any-to-any business document exchange model. The any-to-any business document exchange model is used to exchange documents in any format, such as XML or American National Standards Institute (ANSI) X12 EDI, among different B2B applications. The B2B hub architecture integrates the applications running in companies by using a common message-processing hub. The message-processing hub receives messages from an application, processes it and sends it to another application. Fig.Biz-6.3 shows the B2B hub architecture.

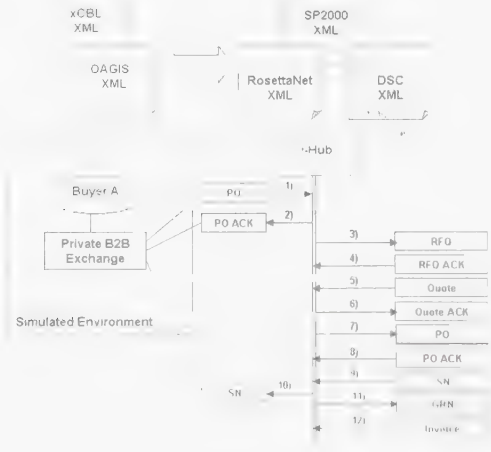


Fig.Biz-6.3

This figure shows the B2B hub architecture in which a buyer company sends messages, such as PO and RFQ, through the B2B exchange to a hub.

Next, we will discuss B2B buy-side and sell-side solutions that provide B2B transactions between B2B trading partners.

B2B Buy-side and Sell-side Solutions

Companies use buy-side and sell-side solutions to perform B2B e-commerce with their partners. A buy-side solution is a group of B2B web applications that run on a buyer company. In addition to setting workflow and internal routing of products, buy-side solutions help perform tasks, such as sending of POs, and acknowledging receipts of invoices. A buy-side solution is used for performing the following tasks in B2B e-commerce:

- Managing the internal workflow of a buyer company by approving material requirement raised by an employee or a department.
- Performing Maintenance Repair and Operations (MRO) services in a buyer company.
- Extracting and comparing data from catalogs of different supplier companies.

A buy-side solution provides the following benefits to the buyer company:

- It reduces the cost of administration.
- It reduces the number of times a purchase cycle is performed.
- It helps manage the stock of products and services required by a buyer company.

A sell-side solution is a set of applications that run on the supplier side and allows a buy-side solution to connect to it directly or through a B2B exchange. A buyer company can obtain complete information related to a product or service sold by the supplier company by using the sell-side solution. A B2B exchange allows a supplier company to publish only limited information, such as product name, product ID, or product price. As a result, the supplier company can customize its sell-side solution to allow a buyer company to directly connect with it and obtain information related to its products and services. Alternatively, a buyer company can obtain supplier information through a channel, such as B2B exchange or e-marketplace.

Now, we will discuss data exchange formats and specifications in detail.

Data Exchange Formats and Specifications

B2B data exchange and transactions are performed through standard data exchange formats used to exchange documents among B2B partners. For a B2B transaction, the B2B partners need to define one or more specific data exchange formats and all the trading partners must be compatible with the selected formats. BZ 2006 supports various data exchange formats for B2B transactions. The data exchange formats supported by BZ 2006 are:

- American National Standards Institute (ANSI) X12
- United Nations Rules For Electronic Data Interchange For Administration, Commerce, and Transport (UN/EDIFACT)
- Open Catalog Interface (OCI)
- Commerce XML (cXML)

Let's look at each format in some detail.

American National Standards Institute (ANSI) X12

American National Standard Institute (ANSI) X12 is based on the Electronic Data Interchange (EDI) standard that transmits a document into different B2B applications. The document transmitted in ANSI X12 format is structured into three levels. These are:

- **Transaction set**: Contains one or more data elements that are used to transfer information from one B2B application to another.
- **Functional group**: Contains one or more transaction sets.
- **Interchange**: Contains one or more functional groups.

A number of segments are required to transmit each document in an ANSI X12 format. These segments provide specific information about the document. The various segments in an ANSI X12 document are:

- **Interchange Control Header (ISA):** This segment acts as a header that represent the start point for an interchange document (Interchange documents allow you to easily store and access your documents online). It also provides information, such as name and ID, about the sender and the recipient of a document.
- **Function Group Header (GS):** This segment acts as a header and represents the start point for a functional group (A functional group contains one or more transaction sets. See introduction to ANSI X12 in this chapter). The information provided by this segment includes code to identify a specific message and a unique reference number assigned to a functional group.
- **Transaction Set Header (ST):** This segments acts as a header and specifies the start point for a transaction set. It also provides information about the type of transaction set and a unique reference number assigned to a transaction set.
- **Transaction Set Trailer (SE):** This segment acts as a trailer that specifies the endpoint for a transaction set. This segment provides the total number of elements in a transaction set and a reference number. This reference number must be the same number as specified in the ST segment.
- **Function Group Trailer (GE):** This segment acts as a trailer that specifies the endpoint for a functional group. This segment also specifies the total number of transaction sets in a functional group and a reference number. This reference number must be the same number as specified in the GS segment.
- **Interchange Control Trailer (IEA):** This segment acts as a trailer that specifies the endpoint for an interchange. This segment also specifies the total number of functional groups in an interchange and a reference number. This reference number must be the same number as specified in the ISA segment.

Table 6.1 shows the structure of a sample ANSI X12 invoice document:

Table 6.1: Structure of a Sample ANSI X12 Invoice Document

Segment	ANSI X12 Format	Sample Invoice Content
ST	ST*810*0001<CR>	

Table 6.1: Structure of a Sample ANSI X12 Invoice Document

Beginning Segment for Invoice (BIG)	BIG*980221*10011 1**E00012345 <CR>	Invoice Date: January 21, 2008 Invoice Number: 100140 Purchase Order Number: E000125 Document Type (not on invoice) DI, CR, DR
Currency (CUR)	CUR*SE*USD <CR>	Currency information (not on invoice)
Name, Address, Geography, N1, N2, N3 & N4	N1*BT*ABC Corporation*92*IT L1 <CR> N3*PO Box 1000 <CR> N4*New Delhi*OR*97124*I NDIAINDIA <CR>	Bill to: ABC Corporation ID Code Qual: 92 (not on invoice) ID Code: ITL1 (not on invoice) Address: PO Box 1000 New Delhi, OR 97124-1000
	N1*ST*ABC Corporation <CR> N3*5300 Ansari Road, Daryaganj <CR> N4*New Delhi*OR*97124*I NDIA <CR>	Ship to: ABC Corporation Address: 5300 Ansari Road, Daryaganj New Delhi, OR 97124
	N1*SF*Acme Co. N3*987G-Block. <CR> N4*Main City*OR*97000 <CR>	Ship From: Acme Co. Address: 987 G-Block, South Ext, New Delhi OR 97000
Admin. Comm. Contact (PER)	PER*AD*Till Acme*TE*5035554 321<CR>	Contact Name: Till Acme Telephone: 503.555.4321
Terms of Sale (ITD)	ITD*01*3*2**10** 30 <CR>	Terms of Sale: 2% 10, net 30 days
Reference	REF*OI*999 <CR>	Reference Invoice number 999
Ship Date	DTM*011*960131 <CR>	Date Shipped 1/31/1996

Table 6.1: Structure of a Sample ANSI X12 Invoice Document

		Quantity	Unit of Measure	Unit Price	Product Desc	Part #
DETAIL (Line Item) Invoice data, IT1	IT1*1*10*EA*5.00 **BP*X1234 <CR>	10	EA	5.00	Widgets	X1234
Product Description (PID)	PID*F *Widgets <CR>	"F" for free form message. Product Description: Widgets				
Reference Number (REF)	REF*GL*123456789*AMT5000 <CR>	Reference Number: General Ledger number (not on invoice) Distribution description: Amount				
Invoice data, IT1 (Line Item Detail)	IT1*1*5*EA*20.00 **BP*Y9999 <CR>	5	EA	20.00	Wombats	Y9999
Product Description (PID)	PID*F ^ Wombats <CR>	Product Description: Wombats				
Reference Number (REF)	REF*GL*123456789*AMT10000 <CR>	Reference Number: General Ledger number (not on invoice) Distribution description: Amount				
SUMMARY Total Invoice Amount (TDS)	TDS*19000 <CR>	Total Due: 190.00 (Implied 2 place decimal point)				
Tax Info (TXI)	TXI*ST*1000<CR>	Sales Tax Due: 10.00				
Carrier Detail (CAD)	CAD*M****ABC Truck <CR>	Carrier Info: ABC Truck				
Special Handling/Freight	SAC*C*D240*3000 *Freight	Shipping and Handling $30.00				
Transaction Totals (CTT)	CTT*2 <CR>					
Transaction Set Trailer	SE*27*000001 <CR>					

Table 6.1 shows the structure of different ANSI X12 documents. Each ANSI X12 document is specified with a specific format.

Table 6.2 shows commonly used formats for different ANSI X12 documents.

Table 6.2: ANSI X12 Document Formats

Document	ANSI X12 Format Specification
Price/Sales Catalog	832
Purchase order	850
PO Change	860
PO Acknowledgment	855/865
Invoice	810
Payment Order	820
Request for Quotation	840
Forecast	830

United Nations Rules For Electronic Data Interchange For Administration, Commerce, and Transport (UN/EDIFACT)

UN/EDIFACT is an EDI standard used to transmit documents among different computer applications. Table 6.3 shows the various UN/EDIFACT document formats:

Table 6.3: UN/EDIFACT Document Formats

UN/EDIFACT Document	Area	Description
PAYORD	Finance	Represents the format of a payment order document.
REQOTE	Trade	Represents the format of a quotation request document.
QUOTES	Trade	Represents the format of a quotation document.
IFTSTA	Transport	Represents the transport status of a PO.
INVOIC	Trade	Represents the format of an invoice.
ORDCHG	Trade	Represents the request to change a PO.
ORDERS	Trade	Represents the format of a PO.
ORDRSP	Trade	Represents the response for a PO. It specifies whether the PO has been completed or not.

Table 6.3: UN/EDIFACT Document Formats

OSTENQ	**Trade**	**Represents the format of a PO status inquiry document. This document specifies the status of a purchase order.**
OSTRPT	**Trade**	**Represents the format of a PO status report document.**
PARTIN	**Trade**	**Represents the format of a buyer information document.**

Documents used in the UN/EDIFACT format are structured into three levels. These are:

- **Interchange**: This level contains one or more functional groups.
- **Functional group**: This level contains one or more messages.
- **Message**: This level represents an ordered series of characters used to transmit information.

Each UN/EDIFACT document contains different segments with specific information about the document. The following segments are used in an UN/EDIFACT document:

- **Service String Advice (UNA)**: This is an optional segment in the UN/EDIFACT document. This segment defines the characters that act as separators and indicators, such as a plus sign or a colon in a message.
- **Interchange Header (UNB)**: This segment represents a header and specifies the start point for an interchange. This segment also specifies information, such as name and ID, about the sender and the recipient of a document.
- **Functional Group Header (UNG):** This segment acts a header and specifies the start point for a functional group. It also provides information, such as code to identify a specific message and the unique reference number assigned to a functional group, about the functional group.
- **Message Header (UNH):** This segment acts as a header and defines the start point of a message. It provides information such as the type of a message and the unique reference number assigned to a message.
- **Message Trailer (UNT):** Message trailer acts as a trailer and specifies the endpoint of a message. This segment provides information such as total number of segments in a message and a reference number. This reference number must be the same number as specified in the UNH segment.
- **Functional Group Trailer (UNE)**: This segment acts as a trailer that specifies the endpoint for a functional group. This segment provides information such as the total number of messages in a functional group and a reference number. This reference number must be the same number as specified in the UNG segment.**Interchange Trailer (UNZ): This** segments acts as a trailer and specifies the endpoint for an interchange. The UNZ segment provides information such as the total number of messages or functional groups in an interchange and a unique reference number. This reference number must be the same number as specified in **the UNG segment.**

Listing 6.1 shows the sample ORDERS file for a UN/EDIFACT document:

Listing 6.1: The Sample ORDERS File

```
1 UNB+UNOB:1+003897733:01:MFGB-PO+PARTNER
  ID:ZZ+970101:1050+00000000000916++ORDERS
2 UNH+1+ORDERS:S:93A:UN
3 BGM+221+P1M24987E+9
4 DTM+4:970101:101
5 FTX+PUR+3++PURCHASE ORDER BEFORE LINE ITEM INSTRUCTIONS
6 RFF+CT:123-456
7 RFF+CR:1
8 NAD+SE+8049P::92++SUPPLIER NAME
9 NAD+BT+B2::92++ABC COMPUTER CORPORATION+P O BOX 92000+HOUSTON+TX+77692000+US
10 NAD+BY+B2::92++ABC COMPUTER CORPORATION
11 NAD+ST+CM6::92++ABC COMPUTER CORPORATION+CCM6 RECEIVING DOCK:20555 SH
249+HOUSTON+TX+77070+US
12 CTA+PD+:CLARETTA STRICKLAND-FULTON
13 CTA+SR+:STEVE 10/19/92
14 TAX+9++++++3-00105-5135-3
15 CUX+2:USD:9
16 PAT+1++1:1:D:45
17 PAT+22++1:1:D:30
18 PCD+12:2
19 TDT+20++++:::AIRBORNE
20 LOC+16+COMPAQ DOCK
21 TOD+2+CC+:::ORIGIN COLLECT
22 LIN+000001++107315-001:BP
23 PIA+1+AA:EC+123456:VP
24 IMD+F+8+:::PART DESCRIPTION INFORMATION
25 QTY+21:10000000:PCE
26 DTM+2:970301:101
27 FTX+LIN+3++LINE ITEM COMMENTS
28 PRI+CON:50
29 UNS+S
30 UNT+29+1
31 UNZ+1+000000000000916
```

Open Catalog Interface (OCI)

Open Catalog Interface (OCI) is a data exchange format used with a web browser to exchange information between a catalog and a buyer. It is implemented in System, Application, and Product (SAP). SAP is an enterprise resource planning (ERP) software capable of integrating multiple business applications, with each application representing a specific business area such as supply chain management, customer relationship management, and supplier relationship management. The documents in the OCI format are divided into two sections, outbound and inbound.

- **The Outbound Section:** The outbound section consists of information, such as catalog URL and login data that is being sent from a buyer application to a catalog application. When a user selects a product catalog from the web browser, the buyer application uses the information such as the URL of the outbound section. The web browser then directs the product catalog from the buyer application to the catalog application and displays the catalog homepage. The information in the outbound section is structured in a specified format, as shown in Table 6.4:

Table 6.4: Formats in the Outbound Section

Formats	Field Name	Field Name Type	Description
Catalog URL	<blank>	Fixed	Specifies the location of a catalog application.
All catalog-specific fields		Variable	Specifies a set of fields, such as request type and password, for a catalog application.
Return URL	HOOK_URL	Variable	Contains the URL of a buyer application.
Caller	-CALLER	Fixed	Represents the data sent by an external catalog.
OK code	-OkCode	Fixed	Contains the transaction code, which indicates that the items are added into the shopping basket. function.
Target	-TARGET	Fixed	Specifies a frame into which a catalog is to return in a frame-based environment.

- **The Inbound Section**: The inbound section consists of information sent by a catalog application to a buyer application. When a user selects a catalog from a web browser, the catalog application generates a document containing the catalog in the HTML format. The catalog application uses the document and redirects it to the web browser to display the data of the buyer application. The catalog in the inbound section is structured in a specified format, as shown in Table 6.5:

Table 6.5: Formats in the Inbound Section

Catalog item	Item name	Description
Description	NEW_ITEM-DESCRIPTION	Represents information about ordered catalog items.
Product master	NEW_ITEM-MATNR	Represents the SAP product master number.
Product Group	NEW_ITEM-MATGROUP	Represents the group number of a SAP product.
Quantity	NEW_ITEM-QUANTITY	Represents the number of ordered catalog items.

Table 6.5: Formats in the Inbound Section

Currency	NEW_ITEM-CURRENCY	Represents the ISO code for currency.
Lead time	NEW_ITEM- LEADTIME	Represents the number of days the catalog item is available.
Vendor product number	NEW_ITEM-VENDORMAT	Represents the product number of a seller catalog item.
Customer-specific field	NEW_ITEM-CUST_FIELD	Represents customer information such as name, address, and so on.
Contract number	NEW_ITEM-CONTRACT	Represents the contract number with the supplier company.
Item of a contract	NEW_ITEM-CONTRACT_ITEM	Represents the number of items for which a contract is signed between a buyer and a supplier company.
Unit of measure	NEW_ITEM-UNIT	Represents the unit of measure for a catalog item.
Price	NEW_ITEM-PRICE	Represents the price of an item in a catalog. The price value of an item must have three digits to the right of the decimal point and can have a maximum of 11 digits to the left of the decimal point.
Vendor	NEW_ITEM-VENDOR	Represents the trading partner number in a B2B procurement system.
Manufacturer code	NEW_ITEM-MANUFACTCODE	Represents the manufacturer code number.
Manufacturer product number	NEW_ITEM-MANUFACTMAT	Represents the product number of a manufacturer catalog product.
Service flag	NEW_ITEM-SERVICE	Represents a flag indicating whether the item name refers to a service or a product.
Quotation	NEW_ITEM-EXT_QUOTE_ID	Refers to an external quotation.
Quotation item	NEW_ITEM-EXT_QUOTE_ITEM	Refers to an external quotation item.

Table 6.5: Formats in the Inbound Section

Product ID	**NEW_ITEM-EXT_PRODUCT_ID**	**Represents a value to identify a specific item in the catalog.**
Description	**NEW_ITEM-LONGTEXT_n:132**	**Defines the description of a product.**

Commerce XML (cXML)

Commerce XML (cXML) is an XML based protocol specifically designed for B2B e-commerce. It is based on XML standards to help exchange documents among different B2B partners. B2B partners use cXML to create different documents, such as electronic catalogs, purchase orders and invoices required for a business process. The following models are used to exchange cXML documents among B2B partners:

- Request-Response model
- One-Way model
- **The Request-Response Model:** In this model, the HTTP protocol is used for transacting cXML documents among B2B partners. In this model, a B2B partner (trading partner A) uses the Request document to send a specific request message to another B2B partner (trading partner B). Trading partner B uses the Response document to send a response of the Request document, to trading partner A. Fig.Biz-6.4 shows how cXML documents are exchanged in the Request-Response model:

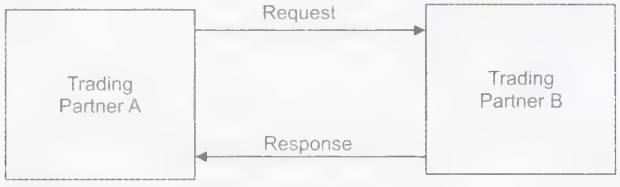


Fig.Biz-6.4

This figure shows the transaction of cXML Request and Response documents between trading partners A and B. The cXML Request document contains a specific request and information to be transacted from trading partner A to trading partner B. Listing 6.2 shows the structure of a cXML Request document.

Listing 6.2: Structure of a cXML Request Document

```
<cXML>
 <Header>
Header information here…
 </Header>
 <Request>
Request information here…
</Request>
</cXML>
```

This listing shows the structure of a cXML Request document, which contains two sections: Header and Request. The Header section contains credential information, such as ID and password of a request user, and the Request section contains a specific request.

The cXML Response document contains the response for the cXML Request document. Listing 6.3 shows the structure of a cXML Response document:

Listing 6.3: Structure of a cXML Response Document

```
<cXML>
<Response>
Response information here
</Response>
</cXML>
```

This listing shows the structure of a cXML Response document containing the Response section, which in turn contains the Response document.

- **The One-Way Model:** In the One-Way model, cXML documents are encoded in a specific protocol for transmitting documents to different B2B partners. Transmitting cXML documents in the One-Way model is not limited to the use of the HTTP protocol. Fig.Biz-6.5 shows how cXML documents are exchanged by using the One-Way model:

Fig.Biz-6.5

This figure shows the transaction of a cXML document from trading partner A to trading partner B. The documents in this model are transmitted in the following way:

- Trading partner A formats and encodes a cXML message that is to be transferred to trading partner B.
- Trading partner A sends the message using a known protocol such as HTTP and does not wait for a response from trading partner B.
- Trading partner B receives the cXML message and decodes it.
- Trading partner B processes the message.

The One-Way model commonly uses the HTTP and URL-FORM-Encoding protocols for transmitting cXML documents. Listing 6.4 shows the structure of a One-Way message document:

Listing 6.4: Structure of a One-Way Message Document

```
<cXML>
<Header>
Header information here…
</Header>
<Message>
Message information here…
</Message>
</cXML>
```

This listing shows the structure of a One-Way message document that contains two sections, Header and Message. The Header section contains credential information and the Message section contains the specific message.

After learning about cXML, let's move ahead and find out how to perform B2B transactions on the Internet.

Open Buying on the Internet (OBI)

Open Buying on the Internet is a proposed standard that provides an open, flexible, and secure framework for B2B transactions on the Internet. It is aimed to be high-volume, low-cost per item transaction and uses a number of security technologies such as a digital certificate to allow orders to be placed and filled securely. The OBI architecture is built on existing standards, such as the X12 850 EDI and X12 855 EDI standards. The OBI architecture contains the following four entities for a transaction process:

- Requisitioner
- Selling company
- Buying company
- Payment authority

In the OBI architecture, a requisitioner at a buying company uses a web browser to access a specific product catalog from the supplier website to place an order. After accessing the product catalog, the requisitioner selects a product and places an order.

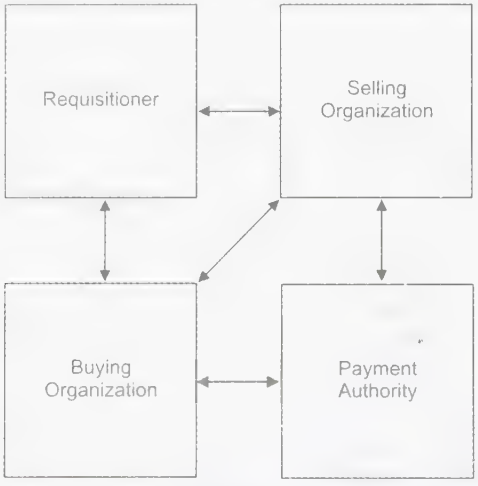


Fig.Biz-6.6

Fig.Biz-6.6 shows the OBI architecture for a transaction process with four entities. These entities are explained as follows:

- **Requisitioner:** The requisitioner represents the person who actually places the order. The requisitioner is affiliated with a buying organization and has access to a desktop machine with a web browser and the Internet. The requisitioner also has a digital certificate, issued by a trusted certificate authority.

- **Buying Organization:** The buying organization represents systems, which support purchasing information. These systems include an OBI server for receiving OBI Order Requests and returning OBI Orders, and also handling the requisitioner profile information, trading partner information, account, and tax status information. The buying organization also maintains contractual relationships with preferred selling organizations.
- **Selling Organization:** The selling organization maintains a product catalog that presents product and price information. This information can be viewed based on organizational affiliation of the requisitioner as specified in a digital certificate. Product and price information reflects the contract with a buying organization. Selling organizations must be able to authorize certain transaction types with the appropriate payment authority.
- **Payment Authority:** Payment authorities provide authorization for the payment presented by the requisitioner. Payment authorities must provide payments to selling organizations and timely debit to the buying organization. Payment authorities may include a variety of financial institutions and if the payment authority is a bulk invoice, selling organizations assume the responsibilities of a payment authority.

Summary

In this chapter, we compared the traditional and the e-commerce methods of doing business-to-business or B2B transactions among different companies. We also discussed the B2B e-commerce process, which has become the preferred way of conducting business transactions through the internet. We discussed the various components involved in the B2B e-commerce process, such as B2B exchange, e-marketplace, B2B partners, B2B hub architecture and various data exchange formats used to perform B2B transactions.

Now that you have an overview of B2B process let's move on to next chapter, where we will learn how to troubleshoot the BizTalk application and handle errors that usually occur during the development stage by using different tools available in BizTalk Server 2006.

Chapter 7

Troubleshooting the BizTalk Applications

In this Chapter

- Health Monitoring
- Common Adapter Errors in BizTalk Server
- Error Handling in BizTalk Server

You may sometimes find that your BizTalk Server displays errors while transmitting messages or while an application is being deployed on BizTalk Server. To resolve these errors, you need to troubleshoot the application for errors. For this, you need a utility to help them identify these errors. In BizTalk, this role is performed by the Health and Activity Tracking (HAT) tool. HAT is used for monitoring the health of BizTalk Server and its applications. By using this tool, you can easily locate the errors in the application, identify their cause, and rectify them by following some simple steps.

This chapter familiarizes you with the working of the HAT tool. This is followed by a detailed discussion on troubleshooting some of the common errors that occur in installing, configuring, and administrating BizTalk Server.

Health Monitoring

The health monitoring tool of BizTalk Server 2006 is used to check or monitor the health of the BizTalk application. The main objective of monitoring the BizTalk application is to find and diagnose the problems occurring in the application and correct them. The health monitoring tool detects these problems, analyzes them and suggests ways of resolving them. Some of the common problems in BizTalk are related to invalid messages, ports, and orchestrations.

The HAT tool is used to debug orchestrations in the BizTalk application. In addition, it also keeps track a message and provides detailed information about it, including its size, its status (that is, whether the message was transmitted successfully or met with failure), its start time (time the orchestration or pipeline started), its end time (time the orchestration or pipeline was completed) and its duration (the time it took for the entire operation to be to completed). Apart from providing this information, the HAT tool also keeps track of schemas, ports and orchestrations used in the BizTalk application. We can also use BizTalk Server Administrator Console, and Event Viewer to monitor the health of the BizTalk application.

We will now discuss HAT in detail in the next section.

Using Health and Activity Tracking (HAT)

The Heath and Activity Tracking (HAT) tool helps you to monitor your BizTalk application. HAT is used for different purposes by different people. System administrators use the HAT tool to track and monitor BizTalk implementation, or view details pertaining to message flow. Business users, on the other hand, use HAT to view, monitor and query tracked data. The main features of HAT are as follows:

- **Tracking Message Flow**: Message flow can be defined as a series of contiguous processing steps taken by a message. The Message Flow view of HAT shows you the sent and received messages of a service instance (pipeline, port or orchestration), with details such as the URL, port, and party used. You can track the message flow and use the information available to troubleshoot errors.
- **Find Message for tracking**: You can use HAT's Find Message feature to search for specific messages. HAT can retrieve messages by using either data or system information. After finding the message, you can view the information for errors and troubleshoot the same.
- **Archived and Live Data for troubleshooting**: Archived data is the backup of data that is no longer used. You can analyze archived data using archiving views to check past data related

to both your business and your system. Live data, on the other hand, is the currently active data stored on the BizTalk Tracking database and shown by views. Any error in Live Data can be rectified by using Archived Data.

- **Orchestration Debugger**: HAT also enables real-time debugging of orchestrations. With the help of this feature, you can set breakpoints to a class or suspended orchestration, view both offline and live data, and view all the steps to design the orchestration again. A breakpoint is a stopping or pausing place of an orchestration that is used for debugging the orchestration.

You can track the overall performance of an orchestration with the help of HAT. You can view a message's technical details such its status, its start time, its end time, the number of sent or received messages, and the queries executed on BizTalk Server Database. You can use the following techniques or options to monitor the BizTalk application with HAT:

- The Find Messages technique or option
- The Query Builder technique or option
- The Message Count in Past Week technique or option
- The Message Count technique or option
- The Message Received in Past Day technique or option

Let us now discuss these one by one in detail.

The Find Message Technique

This technique is used to find out which messages are running under which applications. It displays information regarding the service name (the project name, which is **CustomerOrder** in our case), the schema name (the schema used in the project), and the port name (the Send or Receive ports of the project). Fig.Biz-7.4 displays the message report of the **CustomerOrderFlatFile** schema used in the **CustomerOrder** project.

You can use the **Find Message** technique by following these steps:

1. *Click* **Start→All Program→ Microsoft BizTalk Server 2006→Health and Activity Tracking** to open the **Health and Activity Tracking** window, as shown in Fig.Biz-7.1.

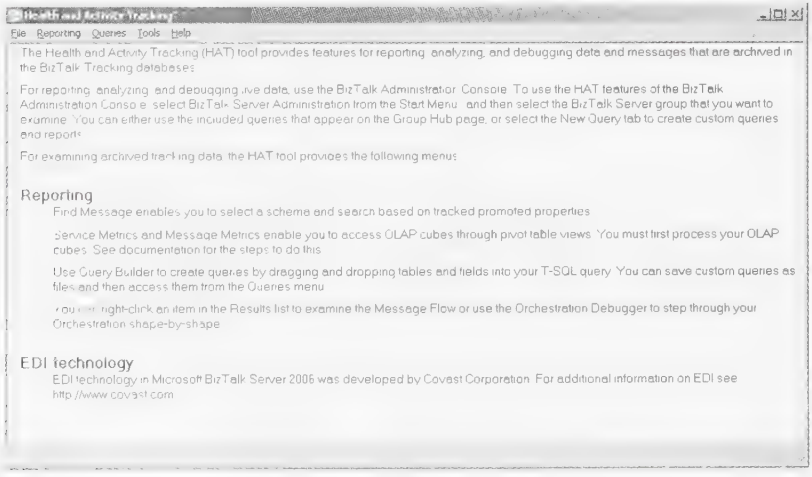


Fig.Biz-7.1

2. *Select* the **Find Message** option from the **Reporting** menu. This displays the **Find Message View** screen, as shown in Fig.Biz-7.2.

Fig.Biz-7.2

3. *Click* the **Schema** button, as shown in Fig.Biz-7.2, and select the **http://CustomerOrder.CustomerOrderFlatFile** schema from the **Schema Selection** dialog box, as shown in Fig.Biz-7.3.

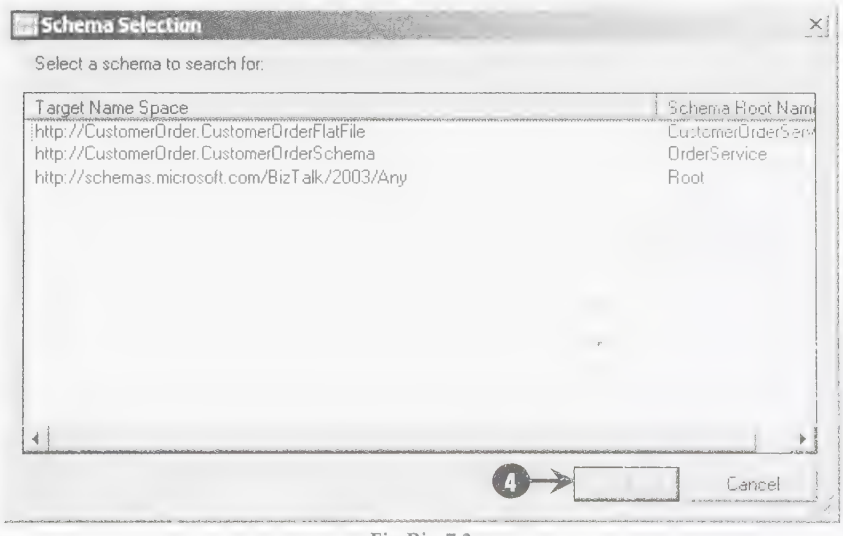


Fig.Biz-7.3

In our case, Fig.Biz-7.3 displays only two schemas of the **CustomerOrder** project (apart from the default schema created by BizTalk Server), which we created in Chapter 4.

4. *Select* the **CustomerOrderFlatFile** schema from the **Schema Selection** dialog box and *click* the **OK** button to open the **Find Message View** screen, as shown in Fig.Biz-7.4.

Selecting the **CustomerOrder.CustomerOrderFlatFile** schema (Fig.Biz-7.3) activates the **OK** button.

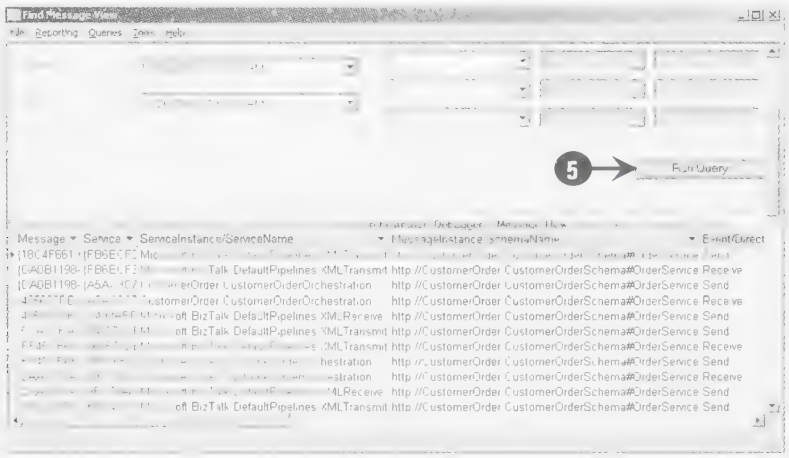


Fig.Biz-7.4

To know more about the CustomerOrderService schema, refer to the 'Adding a Flat File Schema' section of Chapter 4.

5. Now, *click* the **Run Query** button (Fig.Biz-7.4). This will display a list of schema messages sent or received during the dates specified in the **From** and **Until** drop-down lists (Fig.Biz-7.4).

6. The list of schema messages shown in lower part of the **Find Message View** screen (Fig.Biz-7.4) are those that we recently used in the **CustomerOrder** project. *Right-click* any message whose message flow you want to view and *select* the **Message Flow** option from the context menu to open the **Message Flow** screen (Fig.Biz-7.5). Details of the selected message are displayed on your screen (Fig.Biz-7.5)

Fig.Biz-7.5

The Query Builder Technique

You can use the **Query Builder View** technique if you want to retrieve only specific information about a message, such as its size and state, unlike the **Message Flow View** technique, in which all information of a message is displayed. Take the following steps to view a message in Query Builder View:

1. *Open* the **Health and Activity Tracking** tool and *select* the **Query Builder** option from the **Reporting** menu. The **Query Builder View** screen appears, as shown in Fig.Biz-7.6.

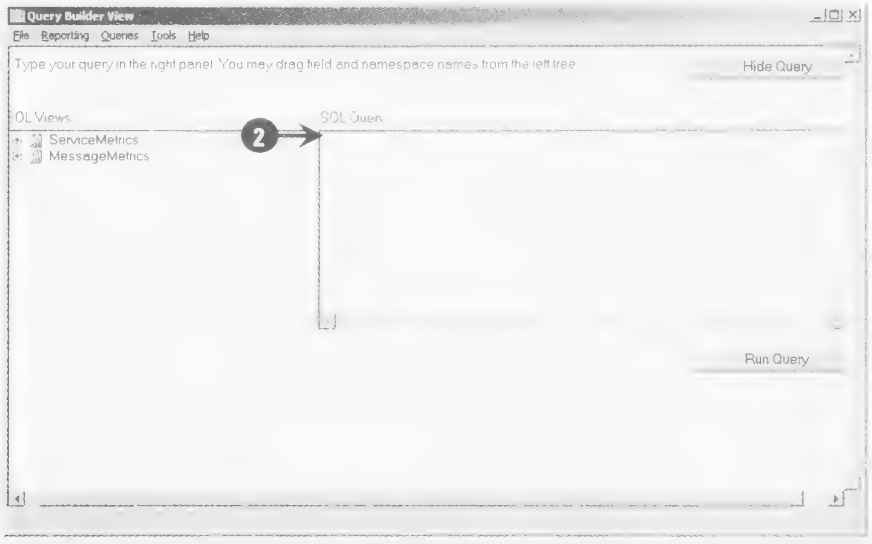


Fig.Biz-7.6

2. Write the query in the **SQL Query** box if you are familiar with its syntax or create an SQL query by simply dragging the various fields in the **ServiceMetrics** or **MessageMetrics** nodes onto the **SQL Query** box.

Message Metrics or Service Metrics node is used to view message details in different ways according to the fields available in these nodes. Various fields available in these nodes are as follows:

- **Instance/State**: This field shows the current state of the message instance.
- **ServiceInstance/State**: This field shows the status of a message, such as either Completed or Terminated.
- **Start time**: This field shows the time the orchestration or pipeline started.
- **End time**: It shows the time the orchestration or pipeline was completed.
- **Duration**: This field shows the time it took for the entire operation to be completed.

In our case, we have selected only the **ServiceInstance/State** field in the **Query Builder** technique, shown encircled in Fig.Biz-7.7. This means that we are viewing only those messages whose status is completed (that is, only successful message will be displayed).

3. After writing the query, *click* the **Run Query** button, as shown in Fig.Biz-7.7. This displays the result of the query in the lower portion of the **Query Builder View** screen (Fig.Biz-7.7).

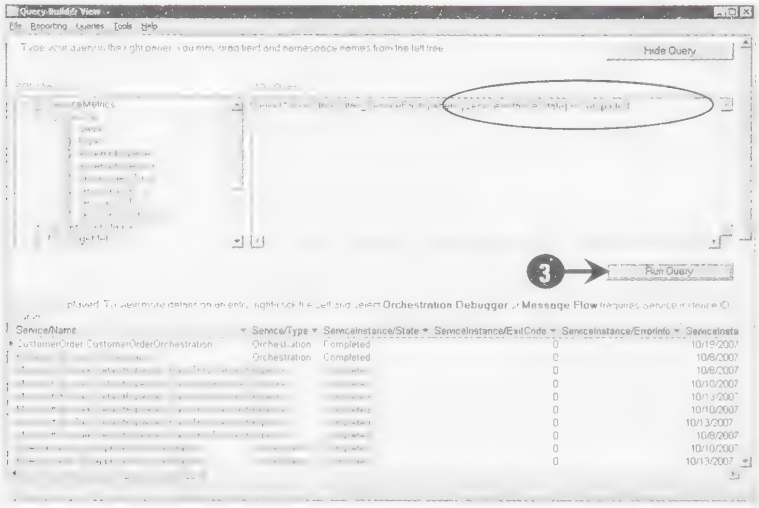


Fig.Biz-7.7

Now, if you want to view the message flow of an orchestration, for example **CustomerOrder.CustomerOrderOrchestration**, simply *right-click* the **CustomerOrder.CustomerOrderOrchestration** row and *select* the **Message Flow** option from the context menu.

Message Count in Past Week Technique

Let us now use HAT to display the number of messages received or sent in the past week as well as the number of message transmission failures that have occurred in the period. To do this, follow these steps:

1. *Select* the **Message count in past week** option from the **Queries** menu. A security warning is displayed the first time you open a query, as shown in Fig.Biz-7.8.

Fig.Biz-7.8

2. *Click* the **Yes** button (Fig.Biz-7.8). This will display the number of received, sent or failed messages in the past seven days on your screen, as shown in Fig.Biz-7.9.

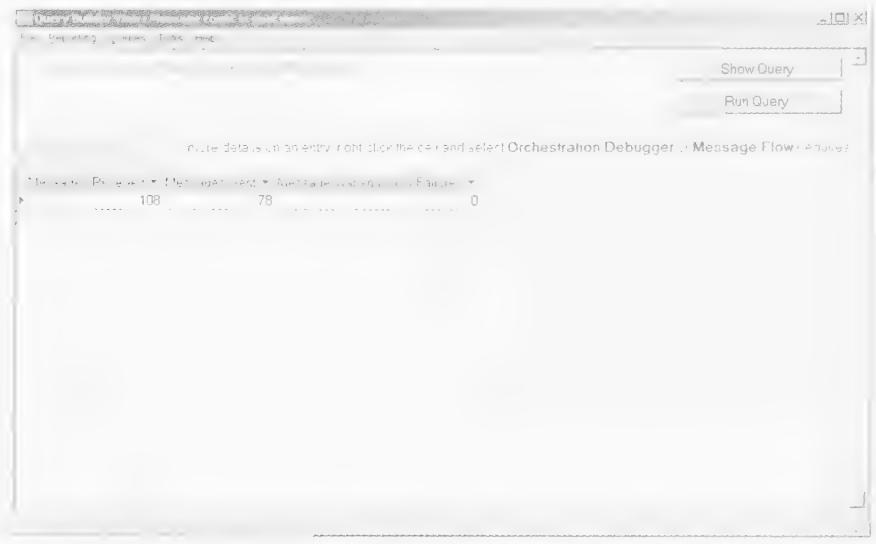


Fig.Biz-7.9

The Message Counts Technique

Use the Message Counts technique to show the number of messages sent and received in the last 1, 2, 7 and 14 days in a table. To view messages by the **Message Counts** technique, select the **Message Counts** option from **Queries** menu. The **Message counts** screen appears, with the number of sent or received messages against the specified days, as shown in Fig.Biz-7.10.

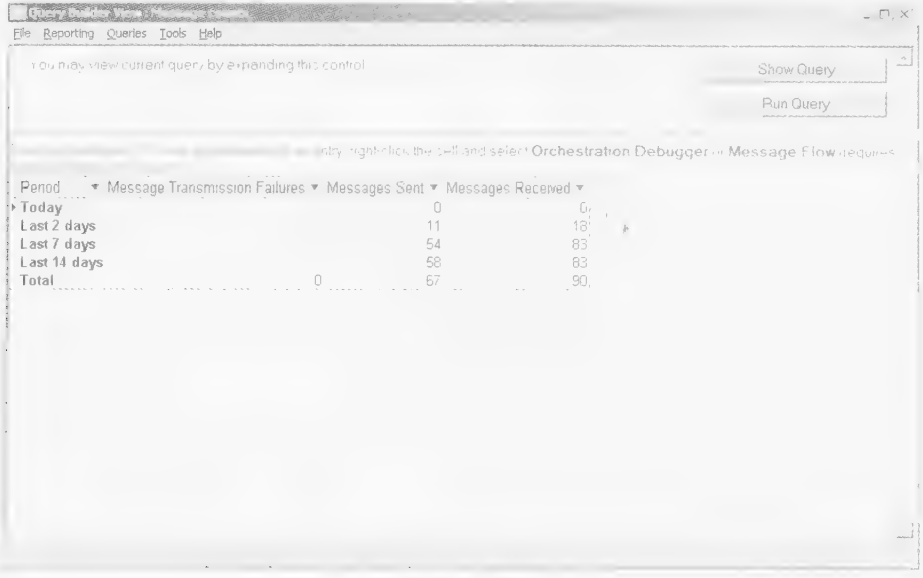


Fig.Biz-7.10

The Message Received in Past Day Technique

You can also view all the messages that the BizTalk application receives in the last 24 hours. To perform this operation, simply select the **Message Received in Past Day** option from the **Queries** menu. The **Messages received in past day** screen appears, as shown in Fig.Biz-7.11. This screen, for example, displays a list of all messages received on 24/10/2007.

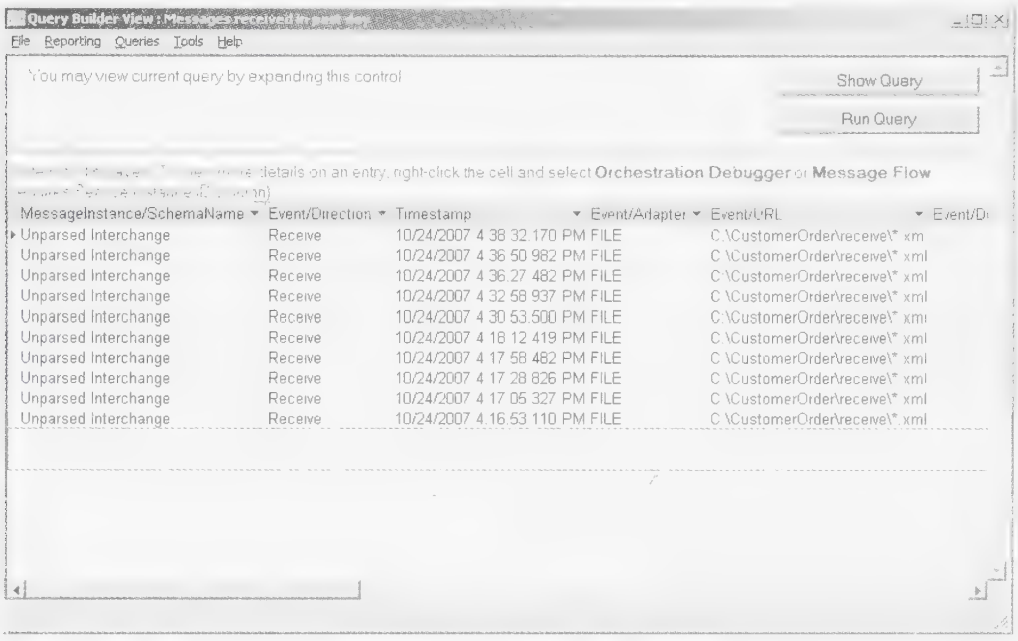


Fig.Biz-7.11

Similarly, to view the messages sent in the last 24 hours, select the **Message Sent in Past Day** option from the **Query** menu. You can also view options including **Most recent 100 service instances**, **Most recent 100 services terminated with errors**, **Recent service instances**, and **Services running longer than 24 hours** by selecting the respective options from the **Query** menu of the **Health and Activity Tracking** tool.

After selecting the Message Counts and Message Received in Past Day options, the warning message appears on your screen as shown in Fig.Biz-7.8. Just click the **OK** button to proceed. You can also debug the application by using the Orchestration Debugger, which you can access by right-clicking the message (in HAT) and selecting Orchestration debugger from the context menu. You also need to set the breakpoints in the Orchestration Debugger for debugging the Orchestration. Breakpoints can be set by using the F9 key on the keyboard.

You now know about the various views in the Health and Activity Tracking (HAT) tool that you can use to monitor as well as check the various components of the BizTalk application. With this, we move to a new section in which we will discuss the common adapter errors in BizTalk Server 2006 and learn how to troubleshoot them. This is important as information is transferred from one location to the other through adapters.

Common Adapter Errors in BizTalk Server

Accessing data using the BizTalk Server may sometimes display error message such as "File adapter not found" or "Receive location is not accessible". These errors occur when some service related to the BizTalk application is not running or the wrong path is specified in the receive location at the time of BizTalk messaging. You need to have some idea of the common errors associated with adapters in BizTalk, along with their possible causes and solutions. Table 7.1 lists some of these errors and also provides ways to troubleshoot them.

Table 7.1: File Adapter Errors

Error	Cause	Solution
File receive adapter is not able to access the specified receive location.	Receive adapter's specified path does not exist. Adapter is unable to read or write the file at the specified location due to restricted rights. File names at the receive location contain more than 256 characters.	Give the existing path or create the specified path location. Ensure that the path of the file is accessible so that it can read or write from the specified receive location. Ensure that the file names in the receive location do not contain more than 256 characters.
Files at the receive location cannot be read by the File adapter.	Files at the receive location are system files. File receive adapter does not have permission to read or write the received file. File names contain more than 256 characters.	Ensure that the file is not a system file. Ensure that the file receive adapter has the read/write permission. Ensure that the file names do not contain more than 256 characters.
File send adapter has problems in sending messages.	File send adapter is unable to access the directory from which files need to be sent because the directory at the specified location does not exist. File send adapter does not have the permission to write the file on the destination location. File to which the message needs to be written at a destination location is read-	Change the path of the specified location or create the directory at that location. Assign the write permission to the file to which the message is to be written through the File send adapter. Assign write permissions for the file at the location to which the message is to be written by the File send adapter.

Table 7.1: File Adapter Errors

Error	Cause	Solution
	only. File to which the message needs to be written at a destination location is a system file.	Make sure that the file at the specified location is not a system file.
File send adapter's message sending speed is very slow.	The Allow cache on write property of the File send adapter is set as False by default.	Change the Allow cache on write property to True.
Files from the specified location cannot be stored or accessed, as these require a username and password.	File send adapter is unable to set the user name and password.	Map a network drive to the location at which the files are to be stored, set the username and password, and then use the mapped network drive in the Send port address.

You now have a general idea of common adapter errors encountered in the BizTalk application and also know how to fix them. Next, we discuss common errors that may crop up while installing, configuring and administering BizTalk Server. We will also talk about the errors you may encounter while implementing the SSO service.

Error Handling in BizTalk Server

Errors can occur at various levels of BizTalk Server. These may come up at the time of:

- installing BizTalk Server
- Configuring BizTalk Server
- Implementing the Enterprise Single Sign-on service
- Administrating BizTalk Server.

Let us now discuss these errors in some detail, along with their solutions.

Handling Errors during BizTalk Server Installation

Installation errors can occur when the software requirements are not met fully while installing BizTalk Server 2006 onto your system. Installation errors can also occur if another application or service is using the software required to install BizTalk Server. Table 7.2 shows the errors that can occur at the time of installing BizTalk Server, along with their solutions.

Table 7.2: BizTalk Server Installation Errors

Error	Solution
Visual Studio .NET is unable to open during the installation of BizTalk Server.	Close the Visual Studio .NET window.
Several Dynamic Link Library (DLL) files are not loaded properly when installing BizTalk Server using Remote Desktop Terminal Services.	Log out of the terminal session (remote desktop computer), and log on again after you install and configure BizTalk Server.
User is unable to install BizTalk Server due to the following error message: Error 1303: The installer has insufficient privileges to access this directory < BizTalk Install directory>\httpreceive. The installation cannot continue. Log on as administrator or contact your system administrator.	To solve the error, do the following: 1. Restart IIS (Internet Information Services). 2. Delete the BizTalk Server's installation folder. 3. Run the BizTalk Server Setup again.
User is unable to install BizTalk Server due to the following error message: "Unable to connect to registry on server (machineName), or you are not a member of the OLAP Administrators group on this server"	Create the user account used by BizTalk Server to connect SQL to the OLAP administrator group. If the problem persists, you need to reinstall SQL Server and add the user account to the OLAP administrator group.

Handling BizTalk Server Configuration Errors

Errors can also occur while configuring the BizTalk Server on the system. The BizTalk Configuration Wizard helps you to configure the BizTalk Services required to run the BizTalk Server. Table 7.3 shows the errors that can occur at the time of configuring BizTalk Server along with their probable causes and solutions.

Table 7.3: BizTalk Server Configuration Errors

Error	Cause	Solution
User is unable to configure BizTalk Server on a domain controller.	This error occurs when the user specifies the local account for the BizTalkServerApplication host or the BizTalkIsolatedHost host instead of the doman account.	Ensure that the domain account is specified in both the hosts.
The Rule Update and BizTalk Message Queuing services are not starting or configuring properly.	The ports used by BizTalk Server for these services are used by another application or service.	Close the application or service that is using the ports, restart the computer, and run the BizTalk Configuration Wizard again.
The following error message appears at the	Occurs when you open Human Workflow Services Console before	Restart Windows Management Instrumentation

Table 7.3: BizTalk Server Configuration Errors

Error	Cause	Solution
time of configuring BizTalk Server: The selected server does not exist or does not have a valid Human Workflow Services installation.	the process of configuring BizTalk Server is completed.	(WMI).
BizTalk Server is not connected with Web Services.	Occurs when you use the SQL Server 2000 Desktop Engine (MSDE) instance for BizTalk Server databases, and the TCP/IP protocols are not enabled.	Enable the TCP/IP protocols by running the following command at the command prompt: \Program Files\Microsoft SQL Server \80\Tools\Bin\svrnetcn.exe
Windows Management Infrastructure (WMI) is unable to create the SQL login.	The BizTalk Configuration Wizard does not disable the local Windows group option on which you want to configure BizTalk Server in the domain controller.	Change the name of the NTGroupName from <ComputerName>\GroupName to <DomainName>\GroupName in the adm_Host table, which is present in BizTalkMgmtDB database Manager.

Handling Single Sign-On Service Implementation Errors

The Enterprise Single Sign-On (SSO) service in BizTalk Sever enables a user to log on to multiple applications by using a single user name and password. Table 7.4 shows the errors that can occur at the time of implementing SSO in the BizTalk Server, along with their probable causes and solutions.

Table 7.4: Enterprise Single Sign-On Service Implementation Errors

Error	Cause	Solution
The following error message appears when a user tries to connect to the remote SSO server by using the ssomanage – displayaap <application name> command: ERROR: 0X800706BA: The RPC server is unavailable	Occurs when users enter incorrect server information or when the SSO service has failed to run on the remote SSO server.	Make sure that the SSO service is running on the remote SSO server.

Table 7.4: Enterprise Single Sign-On Service Implementation Errors

Error	Cause	Solution
User is unable to access an affiliate application.	The application administrator account associated with the affiliate application is not valid.	Ensure that the administrator account associated with the affiliated application is valid.
BizTalk Sever fails to connect to the SSO server.	The SSO server is not available.	Provide the location of the SSO server on BizTalk Server Administration Console.
User is not able to start the ENTSSO service	The ENTSSO service is not running under a valid account.	Run the ENTSSO service under a valid SSO administrator account and restart the service.

Handling Errors during BizTalk Server Administration

Administering BizTalk Server is required for managing the BizTalk Services and its applications. You can manage or administer BizTalk Sever by using BizTalk Administration Console. Table 7.5 lists the various errors that can occur while administering BizTalk Server, along with their probable causes and solutions.

Table 7.5: BizTalk Server Administration Errors

Error	Cause	Solution
User is unable to delete a host from the computer running BizTalk services.	User is not a member of the Windows Administrator group on the local computer.	Make the user a member of the Windows Administrator group.
User is unable to start a host instance on the remote computer.	User enters invalid details in the service account for which BizTalk Server is running or the service account is not granted service rights.	Grant the user service rights for this service account on the remote location.
User is unable to delete a MessageBox database from the computer running BizTalk services.	The cache refresh interval has not expired. MessageBox database contains incomplete service instances. MessageBox database contains unprocessed tracked data.	Disable the new message publication from the MessageBox database and wait until the cache refresh interval expires. If the MessageBox database has suspended service instances, locate these instances using the Health and Activity Tracking (HAT) tool and terminate them.

Table 7.5: BizTalk Server Administration Errors

Error	Cause	Solution
		If the MessageBox database contains unprocessed tracked data, execute the following script: Cscript ForceDeleteMsgBox.vbs < MessageBox Database name>
User is unable to delete database objects associated with a host instance.	User deletes a host instance from the computer running BizTalk services and then tries to delete the associated database objects.	Locate the services related to BizTalk Server on the computer running BizTalk services and stop the services.
User is unable to start the BizTalk Administration Console.	Occurs when the user reinstalls or reconfigures BizTalk Server. After reinstalling or reconfiguring, the end user who wants to start the BizTalk Administration Console is not the member of the Windows Administrator group on the local computer. The user reinstalls or reconfigures BizTalk Server and starts the BizTalk Administration Console without becoming a member of member of the Windows Administrator group on the local computer.	Add the end user as a member to the Windows Administrator group.

This concludes the chapter. What follows next is a short summary of the chapter.

Summary

In this chapter, you have learned how to use the Health Monitoring tool (HAT) to check and monitor the health of BizTalk Server 2006. You have also learned about the common errors in the BizTalk Server related to adapters as well as those encountered during installation, configuration and administration of the BizTalk Server.

By now, you must have a good knowledge of the various components of BizTalk Server 2006 and feel comfortable enough to develop BizTalk applications of your own in .Net. You are also familiar with business rules and policies and know how to create them through the Business Rule

Composer. Moreover, you have a good background of B2B processes that will help you to understand how business is conducted in today's competitive environment. Last but not the least, you know the common errors that you may face while working with BizTalk Server. We hope that you will use the knowledge you have gained from this book profitably to boost your career or professional goals. Best of luck!

Developing BizTalk 2006 Applications

Developing BizTalk 2006 Applications IN SIMPLE STEPS is a book that helps you to develop Custom BizTalk Applications in a precise and complete way. It offers the reader a cutting edge in the field of BizTalk 2006. An easy to understand style, lots of examples to support the concepts, and use of practical approach in presentation are some of the features that make the book unique in itself. The text in the book is presented in such a way that it will be equally helpful to the beginners as well as to the professionals.

The book covers:

- Features of BizTalk 2006.
- Usage of BizTalk Orchestration for performing any business process.
- The development of Custom BizTalk Applications.
- Usage of Schemas, which includes simple and flat file Schemas.
- How to do mapping of XML files.
- Implementing business rules using Business Rule Composer.
- Testing Deployed Custom BizTalk Applications using BizTalk Server 2006 Administration Console and BizTalk Explorer.
- Business process overview.
- Troubleshooting and monitoring BizTalk Applications.

IN SIMPLE STEPS

Published by:

dreamtech PRESS

Dreamtech Press
19-A, Ansari Road, Daryaganj,
New Delhi-110002
Tel.: 91-11-23284212, 23243075
Fax: 91-11-23243078
Email: feedback@dreamtechpress.com

ISBN 10: 81-7722-857-9
ISBN 13: 978-81-7722-857-1

9 788177 228571

SPECIAL INDIAN PRICE

Rs. 179/-

International Price $9.99